4

Perfect and Imperfect Simulations in Stochastic Geometry

Elke Thönnes

This thesis is submitted for the degree of Doctor of Philosophy

Department of Statistics
University of Warwick
Coventry
CV4 7AL

September 1998

Contents

List of Figures

List of Tables

Summary

This thesis presents new developments and applications of simulation methods in stochastic geometry. Simulation is a useful tool for the statistical analysis of spatial point patterns. We use simulation to investigate the power of tests based on the J-function, a new measure of spatial interaction in point patterns. The power of tests based on J is compared to the power of tests based on alternative measures of spatial interaction.

Many models in stochastic geometry can only be sampled using Markov chain Monte Carlo methods. We present and extend a new generation of Markov chain Monte Carlo methods, the perfect simulation algorithms. In contrast to conventional Markov chain Monte Carlo methods perfect simulation methods are able to check whether the sampled Markov chain has reached equilibrium yet, thus ensuring that the exact equilibrium distribution is sampled. There are two types of perfect simulation algorithms, Coupling from the Past and Fill's interruptible algorithm. We present Coupling from the Past in the most general form available and provide a classification of Coupling from the Past algorithms. Coupling from the Past is then extended to produce exact samples of a Boolean model which is conditioned to cover a set of locations with grains. Finally we discuss Fill's interruptible algorithm and show how to extend the original algorithm to continuous distributions by applying it to a point process example.

Acknowledgements

Pursuing a PhD-degree would not be possible without the support of colleagues, friends and family. I thank all who have made it an enjoyable and worthwile experience. I would like to thank my supervisor Wilfrid Kendall for unceasing encouragement and continual support. It was a pleasure and good fortune to work with him. I am also grateful to Marie-Colette van Lieshout whose supervision in my first year made the start of my PhD so much easier. I would also like to thank her for inviting me to visit CWI, Amsterdam. I would like to thank for all invitations to conferences which have been a great encouragement. I kindly thank the Department of Statistics and in particular my supervisors for their efforts to provide me with funding. The financial support through the EPSRC earmarked studentship and the University of Warwick Graduate award is gratefully acknowledged.

Finally, I am grateful to my parents for teaching me to the value of education and trusting in my abilities. Mi último agradecimiento es para Roberto. Tu has sido un infalible apoyo, extremadamente paciente y amoroso a pesar de que no he estado contigo en cuerpo y alma, hubo mucho tiempo que debí haber pasado contigo pero que no fué posible. Estoy mucho agradecida por toda tu patencia, apoyo y comprension. Siempre estaré agradecida porque tu me has enseñado un mundo mas feliz y amoroso.

Declaration

I hereby declare that this thesis is based on my own research, except when stated otherwise.

Chapter 3 is based on the research report: E. Thönnes and M.N.M Van Lieshout (1997) A comparative study on the power of Van Lieshout and Baddeley's J-function, Research Report 334, Department of Statistics, University of Warwick. Submitted for publication.

The simulation study was set-up and performed by myself. M.N.M van Lieshout checked the simulation code and discussed the results with me. Theorem 3.5 was proved jointly.

Chapter 6 is based on the paper: W.S. Kendall and E. Thönnes (1997) Perfect Simulation in stochastic geometry, Research Report 323, Department of Statistics, University of Warwick. To appear in the Special Issue on Random Sets of the Journal of Pattern Recognition.

The research for Chapter 5 was joint work with W.S. Kendall. However, the Theorem 6.3 was my own work.

Chapter 7 is based on the paper: E. Thönnes (1997) Perfect Simulation of some Point Processes for the Impatient User, Research Report 317, Department of Statistics, University of Warwick. To appear in Advances

in Applied Probability, Stochastic Geometry and Statistical Applications.

1 Introduction

In many areas of science data occurs in the form of geometrical patterns. Stochastic geometry aims to provide suitable mathematical models and appropriate statistical methods which enable the analysis of such data. Many stochastic geometry models are defined on high- or infinite-dimensional spaces which renders any mathematical analysis extremely difficult. However, simulation methods may allow us to examine such models by means of statistical analysis. This thesis presents new developments and applications of simulation methods in the area of stochastic geometry. The title "Perfect and Imperfect Simulation in Stochastic Geometry" classifies simulation methods into two categories. The first class of methods is concerned with simulation techniques which allow the "perfect", that is exact sampling of the stationary distribution of a Markov chain. These methods are discussed in Chapters 5, 6 and 7 of this thesis. Other simulation techniques we summarize as "imperfect" methods. The term "imperfect" is not meant to deprecate these methods but merely indicates that they are not concerned with the exact sampling of the stationary distribution of Markov chains. "Imperfect" methods are used in Chapter 3.

Chapter 2 provides a short introduction into the theory of stochastic simulation and gives a brief overview on concepts and models in stochastic geometry. We also point out the rôle of simulation in the area of stochastic

geometry including spatial statistics. The chapter aims to give some of the theoretical background needed for the subsequent chapters.

An example of how simulation enables the statistical analysis of point patterns is given in Chapter 3. Spatial statistics often uses summary functions like the nearest neighbour distribution function G or the empty space function F to quantify spatial interaction in point patterns. Van Lieshout and Baddeley [145] proposed an alternative summary function, the J-function, which is based on F and G through the ratio

$$J = \frac{1 - G}{1 - F}.$$

The theoretical properties of J were presented in [145, 146]. However, it was not examined how well J is suited for statistical tests for spatial interaction compared to the alternatives F and G. Chapter 3 presents a simulation study which compares the power of tests for spatial interaction based on J with tests based on F or on G. Due to the lack of distributional theory for any of these summary functions, hypothesis testing has to be based on simulation via Monte Carlo tests. The chapter extends the simulation study presented in Thönnes and Van Lieshout [141] as follows. We included a comparison of our results with a similar study by Diggle [33]. For this comparison we present results based on Thomas cluster point patterns, which are not described in [141]. Furthermore, we include the Theorem 3.5 which describes a property of the area-interaction point process.

Some of the point patterns used in the simulation study in Chapter 3 were produced by a perfect simulation algorithm. Perfect simulation methods are the topic of the remainder of this thesis. These are a new generation of Markov chain Monte Carlo (MCMC) methods, that is methods which base the sampling of a distribution on a Markov chain. Chapter 4 is an introductory chapter which gives an overview on Markov chain theory and MCMC methods. After presenting basic concepts in Markov chain theory, we introduce coupling, a method on which all perfect simulation algorithms are based. The final section of Chapter 4 is a brief description of MCMC methods including perfect simulation methods.

Markov chain Monte Carlo methods produce an ergodic Markov chain whose distribution converges to the target distribution. They sample the Markov chain after it has run for a long time and thus produce an approximate sample of the target distribution. One major and notoriously difficult problem is to decide how long the Markov chain has to run until its distribution is a sufficiently close approximation to the target distribution. Perfect simulation methods differ from convential MCMC algorithms in that they ensure that the Markov chain is sampled when it has converged to equilibrium. Thus, they sample the target distribution exactly and solve the problem of deciding how long to run the Markov chain. To date there are two types of perfect simulation methods: *Coupling from the Past* which was introduced

by Propp and Wilson in [111] and the *interruptible algorithm* by Fill [37].

Coupling from the Past (CFTP) is based on the intuition that if we started an ergodic Markov chain in the indefinite past, then at time 0 it would have the equilibrium distribution. Chapter 5 shows how this intuition can be translated into a perfect simulation algorithm. We first present the simplest form of Coupling from the Past, as developed in Propp and Wilson [111], and thus illustrate the basic concepts of the method. In Theorem 5.2 we then present Coupling from the Past in the most general form available to date and prove rigorously that Coupling from the Past in this form samples the equilibrium distribution of the target Markov chain. This gives us the general framework which allows us to classify existing CFTP algorithms in the following section. Although there are various overviews on Coupling from the Past [112, 148], this classification of CFTP methods is new. The final section of Chapter 5 discusses implementational issues of Coupling from the Past. In particular, we illustrate how a user who aborts long CFTP runs might introduce a subtle bias in his sample, the user impatience bias.

Theorem 5.2 gives a rather general framework for CFTP algorithms. It is now of primary interest to develop further methods which exploit this freedom and widen the applicability of CFTP. An example of such a method is the Coupling from the Past algorithm presented in Chapter 6 which produces samples of a conditional Boolean model, see also Kendall and Thönnes

[75]. The Boolean model is the union of a number of primary random sets (or grains) which are located at points (or germs) of a Poisson point process. We condition the Boolean model to cover a set of locations with grains. Such conditional Boolean models are used for example in mining engineering where the conditioning allows to incorporate observed data into the model. Approximate samples of conditional Boolean models can be produced through MCMC methods, see Lantuéjoul [79]. We can base an MCMC algorithm for a conditional Boolean model on a conditional spatial birth-and-death process; in Theorem 6.3, which was not presented in [75], we show that such a conditional birth-and-death process does in fact converge to the target conditional Boolean model. We then develop a CFTP algorithm based on a two-component spatial birth-and-death process which samples the conditional Boolean model exactly. The algorithm differs from previous CFTP methods using birth-and-death processes, such as [73, 74, 95], in two aspects. Firstly, whereas previous algorithms thinned births of a birth-and-death process we use perpetuation of particles beyond death times. Secondly, we define our spatial birth-and-death process not on the state space of the target model but on an augmented state space. We show that this construction still satisfies the requirements of Theorem 5.2, thus illustrating the wide applicability of the Theorem.

The second perfect simulation method, the interruptible algorithm by

Fill [37], is presented in Chapter 7. This algorithm is a form of rejection sampling. The runtime of the algorithm (in terms of Markov transitions) is independent of the state sampled. Therefore user impatience bias cannot occur, the algorithm is interruptible. Fill's method is based on the theory of stong stationary times and strong stationary duality as developed in Diaconis and Fill [31]. We give a short summary of the theory and its relation to Fill's algorithm. The interruptible algorithm was originally developed for discrete state spaces; in Section 7.3 we show how to extend it to a continuous state space by applying it to a point process example. This section is a revised version of Thönnes [140]. The final section of Chapter 7 summarizes further extension of Fill's algorithm.

Perfect simulation is a new and exciting research area in which new developments are made at an amazing rate. While researchers increasingly explore the possibilities of perfect simulation, any overview will neccessarily become outdated within a short time. For example, although Theorem 5.2 is the most general formulation of CFTP at present, we expect generalizations of the Theorem soon. Some of these extensions are discussed in the final Chapter 8 in which we present open questions and further work. Perfect simulation has given us some solutions, but many more challenges remain.

2 Simulation and Stochastic Geometry

This chapter will discuss some background material on the two issues of this thesis: *simulation* and *stochastic geometry*. After a short introduction to stochastic simulation, some basic models in stochastic geometry are presented in Section 2.2. Then, in Section 2.2.4 we describe some theoretical concepts which are used in spatial statistics. Thus we provide a theoretical background for the subsequent chapters of this thesis. The last section of this chapter discusses the rôle of simulation in the area of stochastic geometry, thus illustrating the importance of developing new and improving existing simulation techniques.

2.1 Stochastic Modeling and Simulation

2.1.1 Introduction

A *model* is the quantitative description of a natural phenomenon. The purpose of a model is to give insight into the real world by providing explanations and predictions of natural phenomena.

Predictions can usually not be made with certainty and we may want to take this uncertainty into account by incorporating the possibility of randomness into the model. In this case it is a *stochastic* model which describes a set of outcomes which are weighted according to their probability.

Properties of a stochastic model can be examined either by mathematical analysis or *simulation*. Useful and detailed mathematical results for complex models can often only be obtained by either approximation or by imposing additional assumptions which simplify the model. An alternative is to use the model in an empirical experiment, that is to simulate it. We create realizations and examine these to derive properties of the model. Stochastic simulations are used for a variety of classical statistical methods such as Monte Carlo Tests or Bootstrapping and have also profoundly widened the applicability of Bayesian inference methods.

Stochastic simulation methods are often summarized as "Monte Carlo" methods. The term "Monte Carlo" refers to a city in Monaco known for its gambling casinos. It was used during World War II as a code word for simulation experiments of nuclear collision problems. In the literature the term was first used by Metropolis and Ulam [89] in 1949 and has since become a synonym for stochastic simulation techniques. This thesis considers two types of stochastic simulations: *Monte Carlo tests* in Chapter 3 and *Markov chain Monte Carlo methods* in the following chapters.

As with any scientific experiment care is needed when designing simulations. In particular stochastic simulations require an adequate source of randomness. The use of pseudo-random number generators is common; the following section provides a short overview. For a superb introduction to

stochastic simulation see Ripley [118].

2.1.2 Pseudo-Random Number Generators

A sufficient source of randomness for any stochastic simulation is a sequence of independent and identically distributed random numbers which are uniform on the unit interval. Given such a sequence we can produce realizations of many other distributions using a variety of methods such as inversion, rejection sampling or compositions, see [30, 118].

The generation of random numbers should be fast, should require a minimum amount of memory space and should be reproducible. An efficient solution is to use *pseudo-random numbers.* A sequence of pseudo-random numbers is a deterministic sequence which mimics the properties of a random sequence. Pseudo-random numbers need to share the statistical properties of random numbers in the following sense: if we apply standard statistical tests for the departure of independence and uniformity to a finite part of the pseudo-random number sequence the null-hypothesis should not be rejected more often than expected by chance.

Two types of pseudo-random number generators are in common use: linear congruential generators and shift register generators.

As quoted in [78], the *linear congruential generator* goes back to D. H.

Lehmer [80]. We set

$$X_0 \quad = \quad \text{seed}$$

where the seed is some integer and then define

$$X_i \quad = \quad (aX_{i-1} + c) \bmod M, \qquad i \in \mathbf{N}.$$

The pseudo-random number sequence is derived as

$$U_i \quad = \quad X_i/M, \qquad i \in \mathbf{N}.$$

The multiplier a, the shift c and the modulus M are all integers. If $c = 0$ we call the generator multiplicative, otherwise mixed. In a vector $(X_0, \ldots, X_M)$ of pseudo-random numbers not all of the $M + 1$ values will be distinct. Let k be the smallest positive integer such that $X_i = X_{i+k}$ then we call k the *period* of the generator. The period of a generator, which can be at most M, should be as large as possible; the literature [82, 118, 149] recommends a period of 2^{30} or larger.

Usually the modulus M is chosen such that the evaluation of the pseudo-random numbers is possible without a complicated division algorithm and thus is easy and fast. For a digital computer, a power of 2 enables an easy evaluation by simple bit-shift algorithms. After deciding on M the multiplier a and the shift c are then chosen to ensure maximal period. Note that it is essential that the evaluation of X_i, $i \in \mathbf{N}$, is implemented with no round off errors; only then are the properties of the generator guaranteed.

The *shift register generator* in the binary case is based on bits b_i. Define

$$b_i \quad = \quad (a_1\, b_{i-1} + \ldots + a_d\, b_{i-d}) \bmod 2, \qquad i \in \mathbf{N},$$

where $a_1, \ldots, a_n$ are binary constants. From b_i, $i \in \mathbf{N}$, we can derive a pseudo-random number in different ways. For example, Tausworthe [139] proposes the *digital multistep method*:

$$U_i \quad = \quad \sum_{s=1}^{L} 2^{-s}\, b_{in+s}, \qquad i \in \mathbf{N},$$

that is U_i is the binary expansion of a block of bits of length L, each block of bits being n steps apart from the next block of bits.

The *generalized feedback shift register generator* was developed by Lewis and Payne [83] and derives U_i from the L-bit integer composed of non-consecutive bits

$$U_i \quad = \quad b_i\, b_{i-l_2} \cdots b_{i-l_L}\, 2^{-L}, \qquad i \in \mathbf{N},$$

for $l_1, \ldots, l_l \in \mathbf{N}$.

An overview on pseudo-random number generators can be found in [78] or in [104].

All the simulations in this thesis use either the pseudo-random number generator provided by the C Library `stdlib.h` or the programming language S. We chose the function `drand48()` from the C Library, which is a linear congruential generator with parameters

$$M = 2^{48}, \qquad c = 1$$

and

$$a = (273673163155)_8 = 2 \times 8^{11} + 7 \times 8^{10} + 3 \times 8^9 + 6 \times 8^8 + 7 \times 8^7$$
$$+3 \times 8^6 + 8^5 + 6 \times 8^4 + 3 \times 8^3 + 8^2 + 5 \times 8 + 5$$

The generator `drand48()` can be initialized using the C Library function `seed48()`. This function sets the seed X_0 equal to its argument.

The programming language S uses a combination of a linear congruential generator and a shift register generator, see [8].

2.2 Stochastic Geometry

This section gives a brief introduction to those stochastic geometry models which will be used in the following chapters. It starts by describing discrete models, Markov random fields, which originate from problems in statistical physics. In Chapter 5 and Chapter 7 we will introduce a new generation of Markov chain Monte Carlo methods, so-called perfect simulation methods. These methods were first developed for Markov random fields.

The continuous analogues of Markov random fields, Markov point processes, are presented in Section 2.2.2 on point processes. A perfect simulation method for a point process example is developed in Section 7.3. Chapter 3 presents summary functions which can be used to examine point process samples.

Point process models can be extended to produce random sets, two of such models are discussed in the Section 2.2.3 on object processes. In Chapters 5 and 6 we will show how these two object processes can be sampled.

Section 2.2.4 describes theoretical concepts which provide some background for spatial statistics, more details on spatial statistics can be found in Chapter 3.

2.2.1 Markov Random Fields

A *Markov random field* is defined on a finite index set V of sites or vertices, equipped with a neighbourhood system $\partial = \{\partial(v) : v \in V\}$.

Definition 2.1 (Neighbourhood system) *A collection ∂ of subsets of V is called a neighbourhood system, if it satisfies:*

1. $v \notin \partial(v)$ *and*
2. $v \in \partial(w)$ *if and only if* $w \in \partial(v)$.

The most commonly used neighbourhood system is the 4-neighbourhood on a square lattice, where the neighbours are the sites at Euclidean distance one, see Figure 1 for an illustration.

The sites of a Markov random field take values x_v in a finite state space S, thus the vertex set $x = (x_v)_{v \in V}$ takes values in S^V. Let X_v denote the random variable describing the state of site $v \in V$.

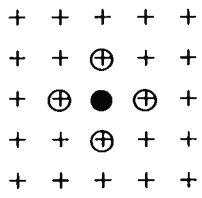

Figure 1: A 4-neighbourhood: The neighbours of • are ∘.

Definition 2.2 (Markov random field) *An S^V-valued random element with distribution π is a* Markov random field *with respect to the neighbourhood system ∂ if*

$$\pi\big(X_v = x_v \;\big|\; X_w = x_w,\ w \neq v\big) = \pi\big(X_v = x_v \;\big|\; X_w = x_w,\ w \in \partial(v)\big). \quad (1)$$

Thus the state at site v depends on the configuration of the other sites in V only through its neighbour sites.

An example for a Markov random field are *spin systems.* Here each vertex $v \in V$ is assigned an upward or downward spin, that is +1 or -1 respectively. We call a spin system *attractive* if $\pi(X_v = 1 | X_w, w \in \partial(v))$ is an increasing function of the values of $\{X_w, w \in \partial(v)\}$.

Example 2.3 (Ising model) *The* Ising model *is a spin system with respect to a 4-neighbourhood. An account of the origin of the model is given in [76]. W. Lenz mentioned the idea for the model in a paper from 1920 [81]. He*

proposed the model to his student E. Ising, who used it in his doctoral thesis to explain ferromagnet phenomena, see also [59]. The model describes a collection of interacting magnets possible together with an external magnetic field. Its distribution is given by

$$\pi(x) \quad = \quad \frac{1}{Z} \exp\Big(-H(x)\Big), \qquad x \in S^V. \tag{2}$$

Here $Z = \sum_{x \in S^V} \exp(-H(x))$ *is the normalizing constant or* partition function *and the exponent* $H(x)$ *is the* energy function*:*

$$H(x) \quad = \quad -\frac{1}{KT}\Big[\, J \sum_{v<w} \alpha_{v,w} x_v x_w - Bm \sum_v x_v \,\Big].$$

The constant $\alpha_{v,w} = 1$ *if* v *is a neighbour of* w *and 0 otherwise. The second sum in the energy function is the effect of an* external field *of intensity* B*; the constant* m *describes a property of the material. The* absolute temperature *in the system is given by the constant* T *and* K *is the* Boltzmann factor. *For* $J > 0$ *spins of the same direction have a low energy and thus a high probability of occuring. This is the* ferromagnetic *Ising model which is an attractive spin system. The repulsive case* $J < 0$ *is known as the* anti-ferromagnetic *Ising model.*

A generalization of the Ising model is the *q-state Potts model* [108, 150], where each of the sites is in one of q different states. A further generalization is the *random cluster model* which was introduced by Fortuin and Kasteleyn [40].

Example 2.4 (Random cluster model) *This model is based on an undirected graph, where each vertex v is connected with vertex w by an edge $\{v, w\}$ with probability $p_{v,w}$. A state of a random cluster model is a subset H of the edges of the graph. The probability of observing the subgraph H is proportional to*

$$\Big(\prod_{v,w \in H} p_{v,w} \Big) \Big(\prod_{v,w \notin H} (1 - p_{v,w}) \Big) \, q^{C(H)},$$

where $C(H)$ is the number of connected components in the subgraph and q indicates how favorable it is to have connected components. Given a state of a random cluster model we can derive a state of a q-state Potts model by assigning randomly one of the q states to each connected component of the subgraph H, see [40].

The modern foundations of Markov random field theory go back to Spitzer [132] and Preston [109]. For a detailed discussion on Markov random fields see [76, 149].

Markov random fields are often defined on large lattices, for example a 1024×1024 square lattice. This leads to huge state spaces and thus delivers partition functions which are not computable in closed form. Nevertheless, (approximate) sampling of Markov random fields is possible through Markov chain Monte Carlo methods such as the heat bath algorithm, see Section 4.3. Exact sampling of Markov random fields is often possible through perfect simulation algorithms which are introduced in Chapter 5 and Chapter 7.

2.2.2 Point Processes

Point processes are models for random patterns of points. The term "point process" was probably first used in Palm's 1943 paper [106] on intensity fluctuations in telephone systems, see the historical remarks in [27]. Although a "process" generally describes an evolution in time, point processes are usually not considered to be dependent on time. Palm intitially used the term "process" as he considered point processes on the real line which can be interpreted as modelling events in time. The oldest and analytically most tractable point process is the Poisson point process. The Poisson process was, according to [27], used as early as the late 1870's to discuss counting problems [1, 129]. Poisson point processes play a dominant rôle in stochastic geometry as they are a building block for many geometric models.

In the following we restrict our attention to *finite Markov* point processes, which can be viewed as the continuous analogues of Markov random fields. Detailed surveys on the general theory of point processes can be found in [24, 27, 69, 114].

We start by constructing the space Ω of realizations of the point process. Let $(S, \mathcal{B}, \lambda)$ be a measure space, where $\mathcal{B}$ contains all singletons and λ is a positive, finite and diffuse measure.

For $n \in \mathbb{N}$, let Ω_n be the space of finite point configurations $x = \{x_1, \ldots, x_n\} \subset S$ with n points. Note that the n points are distinct as we excluded

the possibility of multiple points when assuming that λ is a diffuse measure. Furthermore, we regard x as an unordered set of points in contrast to an ordered n-tuple.

Let S^n, $n \in \mathbb{N}$, be the n-fold cartesian product of S with itself and $\omega_n : S^n \to \Omega_n$ be the projection of ordered n-tuples to unordered point configurations with n points. We equip the space Ω_n with the σ-algebra $\mathcal{F}_n = \omega_n(\mathcal{B}^n)$, where $\mathcal{B}^n$ is the product σ-algebra of S^n. Define $\Omega_0 = \{\emptyset\}$, the empty point configuration, which is equipped with the trivial σ-algebra $\mathcal{F}_0$. Now, let $\Omega = \cup_{n=0}^{\infty}\Omega_n$ and $\mathcal{F}$ be the smallest σ-algebra containing each of $\mathcal{F}_n$ for $n \in \mathbb{N}_0$.

Here are two examples which illustrate the set-up:

Example 2.5 *1. Let $S = A$ be a bounded subset of $\mathbb{R}^d$, $\mathcal{B} = \mathcal{A}$ the Borel σ-algebra on A and $\lambda = \lambda_L$ the Lebesgue measure. This provides the set-up for ordinary point processes where λ_L is the intensity of a unit-rate homogeneous Poisson point process.*

2. *Let $(M, \mathcal{M}, Q)$ be a probability space of "marks", then $(S, \mathcal{B}, \lambda) = (A \times M, \mathcal{A} \otimes \mathcal{M}, \lambda_L \otimes Q)$ provides a set-up for a* marked point process *with mark distribution Q.*

Definition 2.6 (Finite point process) *A* finite point process *on S is defined as a random variable taking values in $(\Omega, \mathcal{F})$.*

Now, let $(\Omega, \mathcal{F}, P)$ be the exponential space [18] over S. The probability measure P corresponds to a *Poisson point process* Y with intensity measure λ. This means that the random variable Y has distribution P if

1. Y has n points, that is $Y \in \Omega_n$, with probability $\frac{\lambda(S)^n}{n!}\exp(-\lambda(S))$ and
2. given that $Y \in \Omega_n$, the n points of Y are independent and identically distributed on S according to $\lambda(\cdot)/\lambda(S)$.

Now, consider a finite point process X on $(\Omega, \mathcal{F})$ whose distribution has a density f with respect to the reference measure P. The density f describes the likelihood of a configuration x for the process X compared to a Poisson point process with intensity measure λ. The distribution of X is specified by the following:

1. X has n points, that is $X \in \Omega_n$, with probability

$$q_n = \frac{\exp(-\lambda(S))}{n!}\int_S \cdots \int_S f(x_1, \ldots, x_n) d\lambda(x_1) \cdots d\lambda(x_n)$$

and

2. given $X \in \Omega_n$, the n points have a joint probability density given by

$$f_n(x_1, \ldots, x_n) = \frac{\exp(-\lambda(S))}{n!}\frac{\lambda(S)^n f(x_1, \ldots, x_n)}{q_n}.$$

Assume in the following that $(S, \mathcal{B}, \lambda) = (A, \mathcal{A}, \lambda_L)$ as in Example 2.5.

Example 2.7 (Pairwise interaction point process) *A* pairwise interaction point process *on S has the following density with respect to a homogenous unit rate Poisson process:*

$$f(x) \quad = \quad \alpha\, \beta^{n(x)} \prod_{i<j} g(|x_i - x_j|_d), \qquad x \subset S.$$

Here $n(x)$ is the number of points in the configuration x and $|\cdot|_d$ denotes the Euclidean distance in $\mathbf{R}^d$. The interaction function g is non-negative and real-valued, the parameter β is positive. The normalizing constant of the density f is denoted by α.

If $g \equiv 1$ then f describes a homogeneous Poisson point process with intensity measure $\beta\lambda_L$.

If

$$g(r) = \begin{cases} \gamma & \text{if } r < R \\ 1 & \text{otherwise,} \end{cases}$$

where $0 \leq \gamma < 1$, then f specifies the density of a Strauss process *[136, 70] with interaction range R. As $\gamma < 1$ the Strauss process favours point patterns where only few point pairs are closer than the interaction range; thus the pattern will be regular. If $\gamma = 0$ then f describes a* hard core process *in which points are a minimum distance, the hard core distance R, apart from each other.*

The Strauss process is an example of a *Markov point process* [120]. Similarly to Markov random fields, Markov point processes are defined with

respect to a neighbourhood relation $\sim$, that is a symmetric and reflexive relation on S. The neighbourhood of a set $B \subseteq S$ is then defined as

$$\partial(B) = \{\xi \in S : \xi \sim \eta \text{ for some } \eta \in B\}.$$

Definition 2.8 (Markov point process) *A finite point process is a* Markov point process *with respect to* $\sim$ *if its density* f *is a* Markov function, *that is*

1. f *is* hereditary*: if* $f(x) > 0$ *then* $f(y) > 0$ *for* $y \subset x$;

2. *if* $f(x) > 0$ *then for* $\xi \in S$,

$$\frac{f(x \cup \xi)}{f(x)} \text{ depends only on } \xi \text{ and } \partial(\xi) \cap x.$$

Example 2.9 (Pairwise interaction point process) *Suppose* $\xi \sim \eta$ *if* $|\xi - \eta|_d < R$. *Say* $g(r) < 1$ *for* $r < R$ *and* $g(r) = 1$ *for* $r \geq R$. *Then pairwise interaction processes are Markov point processes.*

Example 2.10 (Area-interaction point process) *An example of a Markov point process where the interaction is not pairwise but of infinite order is the* area-interaction process. *This process was first introduced as the* penetrable spheres model *in [147] and then extended in [5]. It is defined by the density*

$$f(x) \quad = \quad \alpha \, \beta^{n(x)} \, \gamma^{-\lambda_L(x \oplus K)}, \qquad x \subset S. \tag{3}$$

The exponent of γ is the negative value of the volume of the random set $x \oplus K$ where K is a compact set. The symbol $\oplus$ stand for the Minkowski addition which was introduced by Matheron [87]. The random set $x \oplus K$ is the union of congruent compact sets K centered at the points in x that is

$$x \oplus K \quad = \quad \{x_i + r : \; x_i \in x, \; r \in K\}.$$

Figure 2 shows a random set $x \oplus K$ where K is a disc of fixed radius.

The area-interaction density weights the Poisson point process distribution according to an exponential of the volume of $x \oplus G$. If $\gamma < 1$ it favours configurations for which the volume is large and thus the process will produce regular patterns. For $\gamma > 1$ it models aggregated point patterns.

There is a one-to-one correspondence between the density f of a Markov point process and the *Papangelou conditional intensity* [66, 107].

Definition 2.11 (Papangelou conditional intensity) *The* Papangelou conditional intensity *for a finite Markov point process with density f is defined as*

$$\lambda(x, \xi) = \begin{cases} f(x \cup \{\xi\})/f(x) & \text{if } f(x) > 0 \\ 0 & \text{otherwise,} \end{cases} \tag{4}$$

where $\xi \notin x$.

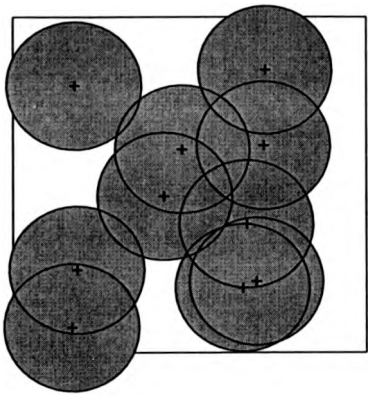

Figure 2: The random set $x \oplus K$, where K is a disc. The location of the points of x are marked as crosses, the random set $x \oplus K$ is shaded.

Informally, the quantity $\lambda(X, \xi)d\xi$ is the infinitesimal conditional probability of there being a point in the infinitesimal region $d\xi$ centered at ξ given the configuration of X outside of this region. Because the Papangelou conditional intensity for Markov point processes depends only on the point ξ and its neighbours in X, it reduces to a computationally simple expression.

Example 2.12 (Papangelou conditional intensity)

1. *For the Strauss process we have* $\lambda(x, \xi) = \beta\gamma^{t(x,\xi)}$, *where* $t(x, \xi)$ *are the number of points in* x *which are neighbours of* ξ.

2. *For the area-interaction process the Papangelou conditional intensity is*

$$\lambda(x, \xi) \quad = \quad \beta\gamma^{-\lambda_L(U(x\cup\xi)\setminus U(x))}.$$

The exponent is the negative value of the amount by which the volume of $U(x) = x \oplus K$ increases when ξ is added to the pattern x.

For most Markov point processes, the normalizing constant of the density is not computable in closed form and thus elementary sampling of these processes is not feasible. Fortunately, approximate sampling procedures are available in the form of Markov chain Monte Carlo methods, see Section 4.3. Some Markov point processes can be sampled exactly using perfect simulation methods, see Chapters 5 and 7.

2.2.3 Object Processes

This section gives a brief description of the random set models we use in Chapters 5 and 6.

The *Boolean model* is one of the simplest examples of a *germ-grain model*, first discussed by Matheron [87] and further developed by various researchers [61, 130] at the École des Mines in Fontainebleu, France. The model is constructed as follows. Suppose $X = \{x_1, x_2, \ldots\}$ is a planar Poisson point process with intensity measure $\lambda(\cdot)$; this is the *germ process*. Let Θ be a random compact set in $\mathbf{R}^2$, the *primary grain*. We assume that

$$\mathbf{E}\,(\lambda_L(\Theta \oplus K)) \quad < \quad \infty \qquad \text{for all compact } K. \tag{5}$$

Suppose $\Theta_1, \Theta_2, \ldots$ is a sequence of independent copies of Θ and assume that this sequence of random sets is independent of X

On each germ $x_i \in X$ we place the random set Θ_i. The Boolean model Ξ is then defined as the random set

$$\Xi \quad = \quad \bigcup_{x_i \in X} \left(x_i \oplus \Theta_i \right).$$

The condition (5) ensures that only finitely many grains $x_i \oplus \Theta_i$ meet any compact set and thus that Ξ is a closed set. For a further discussion of the Boolean model see [87, 130].

The *dead leaves model* was also originally proposed by Matheron [87], then extended and applied to various problems by Jeulin [62, 63, 64]. The dead leaves model is a random tessellation which is defined as the equilibrium distribution of a sequential germ-grain process. Imagine looking down onto a piece of ground which is covered in dead leaves. The leaves on top partially hide the ones below. The resulting image is a superposition of randomly placed leaves: the dead leaves process models this kind of superposition.

Let Θ be a random primary grain, for example a randomly orientated leaf outline. Independent, identically distributed realizations of this grain fall randomly onto a plane at time instants given by a Poisson process of intensity $\theta\, dt$. Grains which appear between time t and time $t + dt$ cover portions of former grains generated at times $u < t$. As a grain is laid down, the boundaries of the new grain is recorded, while the boundaries of older grains which are covered by the new grain are erased. The procedure is illustrated in Figure 14 on page 129, which is taken from [75].

If this process is started at time $-\infty$, then heuristics indicate that at time 0 the dead leaves process will be in statistical equilibrium and the plane will be completely covered, producing a random tessellation of the plane. This tessellation is a realization of the dead leaves model.

2.2.4 Some Theoretical Concepts in Spatial Statistics

Many statistical problems are based on geometrical data such as maps of trees, vegetation patterns, rainfall maps, microscopic images or sections of biological tissues. Often this data can be interpreted as a realization of a point process. Daley and Vere-Jones [27] point to a paper by Cox [23] from 1955 as the first treatment on the statistical analysis of data generated by point processes. Recent monographs on spatial statistics include [35, 69, 117, 119].

An issue of primary interest when examining spatial point patterns is whether the pattern exhibits some kind of spatial interaction. Many spatial statistics aim to detect and quantify such interaction. A pattern which does not exhibit any spatial interaction is called *completely random* and is considered as a realization of a Poisson point process. Thus the Poisson point process often functions as a model for comparison or, in statistical terms, as a null hypothesis in tests for interaction in point patterns. This section introduces some of the theoretical quantities which are commonly used in

spatial statistics; for further details see the simulation study in Chapter 3.

We start with a characteristic of point processes which is the analogue to the mean of a real-valued random variable. Let X be a point process and let $X(A)$ denote the number of points of X in the Borel set A. In the following we call a point of the process an incident in order to distinguish it from an arbitrary point in space.

Definition 2.13 (Intensity measure) *The* intensity measure λ *of* X *is defined as*

$$\lambda(A) \quad = \quad \mathbf{E}\left(X(A) \right) \qquad \textit{for a Borel set } A. \tag{6}$$

Often spatial statistics methods make the assumption that the intensity measure is proportional to the Lebesgue measure. The constant of proportionality is called *intensity* and quantifies the mean number of incidents per unit volume.

Definition 2.14 (Homogeneity) *A point process* X *is called* homogeneous *or* stationary *if any translation of* X *has the same distribution as* X.

For stationary point processes it can be shown that the intensity measure is proportional to the Lebesgue measure.

Statistics for point patterns are often based on properties of a "typical" point of the process. For example, the distance of a "typical" incident to the

nearest other incident is used to detect interaction between incidents. The *Palm distribution* specifies the concept of a "typical" point.

Definition 2.15 (Palm distribution) *Let λ be the intensity measure of X. The* Palm distribution $\mathbf{P}^x$ *of X at $x \in \mathbb{R}^d$ is defined by the equation*

$$\mathbf{E}\big(X(A)\,1_C\big) \quad = \quad \int_A \mathbf{P}^x(C)\,d\lambda(x), \qquad C \in \mathcal{F}. \tag{7}$$

The measure $C(A \times C) = \mathbf{E}(X(A)1_C)$ is the *Campbell measure*, which we assume to be absolutely continuous with respect to λ. Heuristically, $\mathbf{P}^x(\cdot)$ is the conditional distribution of X given that there is an incident at x, the "typical" point. Notice that for a stationary point process the Palm distribution $\mathbf{P}^x$ does not depend on x. A related distribution is the *reduced Palm distribution* $\mathbf{P}^{!x}$.

Definition 2.16 (Reduced Palm distribution) *The* reduced Palm distribution $\mathbf{P}^{!x}$ *of a finite point process X is defined as*

$$\mathbf{P}^{!x}(C) \quad = \quad \mathbf{P}^x(X\backslash\{x\} \in C), \qquad C \in \mathcal{F}. \tag{8}$$

Example 2.17 (Poisson point process) *Consider a stationary Poisson point process with distribution P. Due to* Slivnyak's Theorem *[131] we have*

$$\mathbf{P}^x = \delta_{\delta_x} * P \quad \textit{and thus} \quad \mathbf{P}^{!x} = P,$$

where δ_{δ_x} denotes the distribution of a point process which consists of exactly one incident at x and $*$ denotes the convolution of distributions.

The Nguyen-Zessin formula [103] relates the distribution of a stationary finite Markov point process X to its reduced Palm distribution.

$$\lambda \, \mathbf{E}^{!y} \left(f(X) \right) \quad = \quad \mathbf{E} \left(\lambda(X; y) \, f(X) \right), \tag{9}$$

Here $\mathbf{E}^{!y}$ stands for the expectation with respect to the reduced Palm distribution $\mathbf{P}^{!y}$, the constant λ is the intensity of X and $\lambda(X; y)$ is the Papangelou conditional intensity of X at y. Formula (9) holds under suitable conditions on X for any bounded non-negative measurable function f on the state space of X. A necessary and sufficent condition on X to ensure that the Nguyen-Zessin formula holds is that the reduced Palm distribution of X is absolutely continuous with respect to its distribution

The (reduced) Palm distribution is used to define statistical summary functions such as the *nearest-neighbour distribution function* G which for a stationary planar point process X is given by

$$G(r) \quad = \quad \mathbf{P}^{!0}(X \in b(0, r)),$$

where $b(0, r)$ is a disc of radius r centred at the origin. Chapter 3 presents further summary functions which, like G, quantify the spatial interaction in a point process.

Although a variety of summary statistics for point patterns have been developed, for many of these statistics there is little distributional theory available. Thus statistical inference relies heavily on simulation methods

such as Monte Carlo tests. An example of such a simulation study can be found in Chapter 3.

2.3 Simulation in Stochastic Geometry

The vast majority of models in stochastic geometry are defined on high-dimensional or even infinite-dimensional state spaces. This renders any analytical examination extremely difficult. Many models are based on densities whose normalizing constant cannot be computed in closed form and for which a conventional numerical computation is not feasible. Thus it is not surprising that research within stochastic geometry has been concerned with simulation methods. In particular the early development of sampling methods known as Markov chain Monte Carlo techniques took place in statistical physics and image analysis.

Furthermore, although statistics which summarize properties of models such as point processes are available, there is hardly any distributional theory for these statistics. Thus statistical inference for geometrical data often has to be based on Monte Carlo tests.

Sometimes the development of a simulation algorithm can provide new insights into the processes concerned. For example, Theorem 3.5 in Chapter 3 and Proposition 6.15 in Chapter 6 were motivated by the development of a simulation algorithm.

The following chapters will present examples and new developments of simulation methods and thus will illustrate the important rôle of simulation in stochastic geometry including spatial statistics.

The next Chapter 3 will present a simulation study which examines the power of tests based on a new spatial statistic, the J-function. This function has many useful theoretical properties, see Section 3.3. However, as the sampling distribution of the J-function is unkown, an analytical examination of the power of tests using this function is not possible. Nevertheless, we can assesss the power of these tests by the means of simulation. In Chapter 3 we describe a simulation study which compares the power of tests based on the J-function with tests based on alternative statistics and thus illustrate the usefulness of the J-function when examining point patterns.

The subsequent chapters will focus on new developments in Markov chain Monte Carlo methods. Until recently, Markov chain Monte Carlo methods would generally only produce approximate samples of the target distribution. Perfect simulation methods are a new generation of Markov chain Monte Carlo methods which produce exact rather than approximate samples of the target distribution. We develop two such perfect simulation methods for models in stochastic geometry.

3 A Comparative Study on the Power of the J-function

This chapter concerns the use of simulation within spatial statistics. It is an extended report on a simulation study presented by Thönnes and Van Lieshout in [141]. This study illustrates how simulation enables the analysis of spatial interaction in point patterns. There is hardly any distributional theory available for measures of spatial interaction and thus hypothesis testing is only possible through Monte Carlo tests wich are based on simulation.

Furthermore, simulation allows an assessment of the tools used in spatial statistics. The simulation study presented in this chapter examines the summary function J proposed by Van Lieshout and Baddeley [145] which quantifies spatial interaction in point patterns. Baddeley and Van Lieshout presented the theoretical properties of the J-function in [145] and [146], but did not examine whether the J-function yields more powerful tests for complete spatial randomness than alternative measures of spatial interaction. In this chapter we examine the power of tests based on the J-function and compare it with the power of test based on some alternative measures of spatial interaction in point patterns.

3.1 Introduction

Mapped spatial patterns arise in a variety of contexts, ranging from galaxies in astronomy, the positions of cell nuclei in cytology, maps of tree locations in forestry to the findings of ore in material science.

The statistical analysis of such a mapped point pattern usually begins with a test for *complete spatial randomness* [35, 117]. The hypothesis of complete spatial randomness assumes that the observed point pattern in a window W was produced by a stationary Poisson point process. Recall that a point process is a stationary Poisson point process if

1. the number of points in W with area $|W|$ is Poisson distributed with mean $\lambda|W|$ for some positive constant λ;

2. given that there are n points in W, the locations of these points are independently and uniformly distributed over W.

The second condition amounts to requiring that there be no interaction between the points of the process as the presence of an incident at a certain location has no influence on the locations of the other incidents. This requirement is violated if the observation of an incident at location $x \in W$ increases or reduces the conditional probability of there being other incidents in the vicinity of x. In the former case we have an attractive interaction which yields point patterns which are aggregated (an equivalent term is "clustered"); in

the latter case the repulsive interaction produces regular patterns. Figure 3 shows an example for a completely random, a clustered and a regular point pattern. Notice that a given point pattern may exhibit both attractive and repulsive interaction. For example, the points may be a minimum distance apart from each other but occur in clusters. The aim of a test for complete spatial randomness is to detect such interactions. Throughout this chapter we follow the common assumption that the point pattern was produced by a stationary point process and focus on the detection of interaction between points.

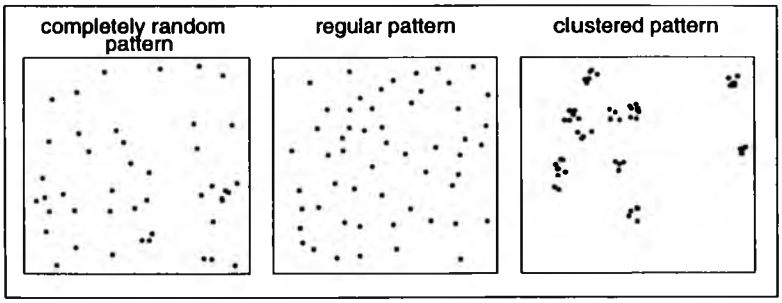

Figure 3: The point pattern on the left is a realisation of a Poisson point process of intensity $\lambda = 50$. The pattern in the middle is a realisation of a hard core process of intensity $\lambda = 50$ and with hard core distance $h = 0.06$. The pattern on the right was produced by a Matérn cluster process with a parent intensity equal to 10, a mean daughter number $\mu = 5$ and radius $R = 0.08$. For a description of the processes see Section 3.4.1.

Low-dimensional summary functions such as the *nearest neighbour dis-*

tance distribution function G or the *empty space function* F are often used as test statistics, see [25, 35, 117, 119] and [133, 134]. Both F and G are distribution functions of distances between points. The empty space function F is the distribution function of the distance from an arbitrary sampling point, for example the origin, to the nearest incident. The function G is the distribution function of the distance from a typical point of the process to the nearest other incident. Recently, Van Lieshout and Baddeley [145] proposed an alternative, the J-function, which compares minimum inter-incident distances to minimum distances from a fixed sampling point. It was hoped that the combination of both types of distances into one statistic would yield a higher power of tests for complete spatial randomness.

For none of the summary functions F, G, and J is there sufficient distributional theory available on which to base statistical inference. Thus an assessment of the power of tests based on these functions is only possible through simulation. This chapter describes a simulation study which compares the power of tests based on the J-function with the power of tests based on F or on G.

The next section will review the summary functions F and G. The alternative summary function J is introduced in Section 3.3. Section 3.4 describes the set-up of the simulation study. The results of this study are presented in Section 3.5 and conclusions are drawn in the final Section 3.6.

3.2 Measures of Spatial Interaction

Recall that throughout this chapter we assume that X is a stationary point process in $\mathbf{R}^2$.

Definition 3.1 (F-function) *The* empty space function F *of* X *is defined as*

$$F(r) = \mathbf{P}\,(\, d(0,X) \leq r\,) = \mathbf{P}\,(X \cap b(0,r) \neq \emptyset\,), \quad r \geq 0. \qquad (10)$$

Here $d(0,X)$ *is the distance from the origin to the nearest incident of* X *and* $b(0,r)$ *is a disc of radius* r *centered at the origin.*

Another common summary function is the following:

Definition 3.2 (G-function) *The* nearest neighbour distribution function G *is defined as*

$$G(r) = \mathbf{P}^{!0}(\, d(0,X) \leq r\,) = \mathbf{P}^{!0}\,(\, X \cap b(0,r) \neq \emptyset\,), \qquad r \geq 0 \qquad (11)$$

where $\mathbf{P}^{!0}$ *denotes the reduced Palm distribution of* X *at the origin, see also Section 2.2.4.*

Example 3.3 (Poisson point process) *Consider a stationary Poisson point process of intensity* λ. *Because of Slivnyak's Theorem [131], see also Example 2.17,* F *and* G *are identical for this process and given by*

$$F(r) \quad = \quad G(r) \quad = \quad 1 - \exp(-\lambda\pi r^2).$$

The summary functions F and G are measures of spatial interaction due to the following reasoning. Suppose we have a clustered point pattern. Clearly, the nearest neighbour distances of incidents will be smaller than expected from a completely random pattern. Thus the G-function of a process with attractive interaction will lie above the G-function of a Poisson point process.

How about the F-function? From the Definition (10) we have

$$F(r) \quad = \quad \mathbb{P}(0 \in X \oplus b(0,r)).$$

Thus $F(r)$ is the area fraction of the random closed set $X \oplus b(0,r)$ which will be smaller the more clustered the point pattern is. It follows that the F-function of a process with attractive interaction will lie below the F-function of a Poisson point process.

Analogously, the G-function of a process with repulsive interaction lies below the G-function of a Poisson point process and vice versa for the F-function. For an illustration see Figure 7 on page 50 and Figure 8 on page 51, which show the empirical counterparts of F and G for the regular and the clustered pattern in Figure 3 and demonstrate their relation to the summary functions of a Poisson point process.

The use of the empty space function F for the analysis of planar point patterns was probably first considered by Ripley in 1977 [115] as a special case of what he called the "test-set method". As reported in [117], the G-

function was first suggested by Cowie in 1976 [21]. Diggle [33, 34] advocated the "refined nearest neighbour analysis", that is the use of both F and G for the analyis of planar point patterns. In 1979 he performed a simulation study [33] which compares a variety of measures of spatial interaction, amongst others the F- and the G-function. He found that for his choice of clustered patterns tests based on F were more powerful. For the regular patterns tests based on G seemed to be more powerful. He also considered combining F and G and proposed a test statistic based on the difference $F(r) - G(r)$ between F and G. However, he found that tests based on the latter statistic were not more powerful but had a power which would lie between the power of tests based on F and tests based on G.

3.3 An Alternative Measure of Spatial Interaction

In [145], the J-function is defined as follows:

Definition 3.4 (J-function) *The* J-function *is given by the ratio*

$$J(r) = \frac{1 - G(r)}{1 - F(r)} = \frac{\mathbf{P}^{!0}(X \cap b(0,r) = \emptyset)}{\mathbf{P}(X \cap b(0,r) = \emptyset)} \tag{12}$$

for all $r \geq 0$ *such that* $F(r) < 1$.

It can be interpreted as the ratio of two survival functions, namely of the distance to the nearest incident from (a) an incident or (b) an arbitrary

sampling point. Thus (12) compares the environment of a typical incident of X to the environment of an arbitrary sampling point.

If $J(r) < 1$, then the survival function of (a) is smaller than that of (b) indicating clustering, whereas if $J(r) > 1$, then the survival function of (b) is smaller than that of (a) indicating regularity. For a Poisson point process F and G are identical, so we then have $J(\cdot) \equiv 1$. Thus, a plot of the (estimated) J-function provides valuable information on the type and strength of interaction between incidents. Furthermore, it can be shown that, in contrast to F and G, the J-function is constant beyond the effective range of interaction. Therefore we can estimate the range of interaction from a plot of an estimate of J. Figure 9 on page 51 shows the estimated J-function for the regular and the clustered point pattern in Figure 3 and illustrates how the J-function measures the type, the strength and the range of spatial interaction.

The functions F, G or J summarize a point pattern using low-dimensional functional statistics, however neither F and G nor J completely determine the distribution of X. Bedford and Van den Berg [9] give an example of a point process that is not Poisson but for which $J(\cdot) \equiv 1$.

J is related to the Papangelou conditional intensity as follows. Consider a stationary point process X of intensity λ whose Papangelou conditional intensity $\lambda(X, \cdot)$ exists. Due to the Nguyen-Zessin formula (9), the J-function

for X can then be expressed in terms of the conditional intensity as shown in [145]:

$$J(r) \quad = \quad \mathbf{E}\left(\frac{\lambda(X;0)}{\lambda} \,\middle|\, X \cap b(0,r) = \emptyset \right).$$

As discussed in Section 2.2.2, for Markov point processes the Papangelou conditional intensity reduces to a computationally simple expression. Thus J can be evaluated for a wider range of point process models than F and G. It also behaves well under independent superposition of point patterns, see [146], a property that is very useful when studying interactions in mapped patterns consisting of different types of incidents.

For formal proofs and further details see [145]. Multivariate extensions of (12) are studied in [146]. Plots of the J-function for Poisson cluster processes can be found in [128], while [4] considers robustness against edge effects caused by incomplete observation of X.

Clearly the J-function has many theoretical advantages over F and G, but how does it perform empirically? Does it lead to more powerful tests than F and G? For which type of patterns is it preferable to F or G? This chapter will give some answers to these questions by means of simulation.

3.4 Simulation Study on the Power of J

3.4.1 The Simulated Processes

In order to compare the power of tests based on J with tests based on F and G, we simulated three different types of non-Poissonian point processes: *cluster processes, hard core processes and area-interaction processes.* We then tested the simulated patterns against a Poisson null hypothesis and estimated the power of these tests by the proportion of rejected patterns. In this section we will describe the simulated processes; the testing procedures will be explained in the next section.

For all of the following processes we obtained samples on a unit square window:

1. *Cluster process:*

A cluster process [27, 102] is derived from a parent process by scattering a cluster of daughter points around each of the parent points. Each parent point has an independent and identically distributed number of daughter points. The locations of the daughter points relative to the parent point are also independent and identically distributed. The union of all daughters then forms the cluster process. We chose two different types of *Poisson cluster processes*, that is cluster processes whose parent process is a stationary Poisson process. For both types of cluster processes each parent point is assigned

a Poisson number of daughter points with mean μ. The first type of cluster process is a *Matérn cluster process* [86], where the daughter locations are uniform on a disc of radius R. The second type of cluster process is a *Thomas field*, where the location of the daugher points is radially normal, centred at the parent point and with standard deviation σ. The clustered pattern in Figure 3 is a realisation of the Matérn cluster process. The simulation of a Poisson cluster process is straightforward, for a description see [134]. To avoid edge-effects we simulated the Matérn process on a square window of side length $1+2R$ and then sampled only those incidents which lay inside the centre unit square. This procedure is called plus-sampling. For the Thomas process we imposed a periodic boundary condition, that is we identified the sampling window with a torus.

2. *Hard core process:*

In a hard core process all points of the process are a minimum distance h, the hard core distance, apart from each other. Matérn proposed two such models in 1960 [86]. We chose the *Type 2 Matérn hard core process* using the algorithm described in [86, 115]. This algorithm applies thinning to a marked Poisson process according to the following rule. Each incident of a Poisson realisation is assigned an independent uniform mark. If two incidents are closer than the hard core distance to each other the point with the smaller mark is deleted. This leads to highly regular patterns; a realisation of this

process can be seen in Figure 3. To avoid edge-effects when simulating the hard core process, we used plus-sampling.

Area-interaction process:

As mentioned in Section 2.2.2, this process is defined by the density

$$p(x) \quad = \quad \alpha \, \beta^{n(x)} \, \gamma^{-\lambda_L(x \oplus K)}$$

with respect to a unit rate Poisson point process on a compact window. Here α is the normalizing constant, $n(x)$ is the number of incidents in x and K is the grain of the process. The parameter $\beta > 0$ influences the intensity of the process. Recall that the parameter γ does not only control the strength but also the type of interaction. For $\gamma < 1$, realisations of the area-interaction process tend to be regular, whereas for $\gamma > 1$ clustered patterns are more likely. Figure 4 shows a realisation of an attractive and of a repulsive area-interaction.

The area-interaction model is one of the few Markov point processes whose exponential family can model satisfactorily both attractive and repulsive interaction. The strength of interaction is very strongly influenced by the choice of the grain K. If the area of the grain is very small, then the exponent $-\lambda_L(x \oplus K)$ will be close to zero leading to very weak interaction. If the area of the grain is very large then $\lambda_L(x \oplus K)$ is close to the area of the

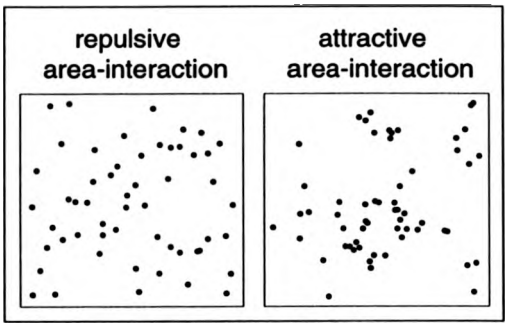

Figure 4: Samples of the area-interaction process. The left sample is a repulsive area-interaction with parameter $g = \ln(\gamma) = -300$, the right sample is an attractive area-interaction with parameter $g = \ln(\gamma) = 250$. Both processes have an intensity of 50 and are based on a square grain of sidelength 0.1.

---

window W. Thus the area-interaction density will be almost proportional to

$$\beta^{n(x)}\, \gamma^{-\Lambda_L(W)}$$

and so close to the density of a Poisson point process. Baddeley and Särkkä are currently investigating how K should be chosen such that the area-interaction process produces patterns with sufficiently strong interaction.

The normalizing constant α of the area-interaction density is not computable in closed form, so we have to use a Markov chain Monte Carlo (MCMC) method to produce samples. We used the algorithm outlined in Kendall [73], coded in his C program `Perfect`. To simplify computations this program uses a square for the grain K and assumes empty boundary

conditions. The algorithm is based on the *Coupling from the Past* idea of Propp and Wilson [111] which, in contrast to conventional MCMC methods, allows for exact rather than approximate samples. Coupling from the Past is discussed in Chapter 5; a description of Kendall's algorithm can be found in Section 5.4.3. The development of this algorithm is an example of how simulation can provide new insights into stochastic geometry models. While developing the program `Perfect`, Kendall conjectured the following Theorem.

Theorem 3.5 *Suppose there is a subset of points y in an area-interaction realisation x such that $y \oplus K = x \oplus K$. Then the points of x which are not in y are a completely random pattern on $(y \oplus K) \ominus K$.*

Theorem 3.5, which is illustrated in Figure 5, can be proved as follows:

Proof: We have

$$\begin{aligned} x \oplus K &\subseteq y \oplus K \\ \iff x_i \oplus K &\subseteq y \oplus K \qquad \text{for all } x_i \in x \\ \iff x_i &\in (y \oplus K) \ominus K \quad \text{for all } x_i \in x \\ \iff x &\subseteq (y \oplus K) \ominus K, \end{aligned}$$

where $\ominus$ is the Minkowski-subtraction [87] that is

$$A \ominus B = \bigcap_{y \in B} \{x + y : x \in A\} = (A^c \oplus B)^c. \qquad (13)$$

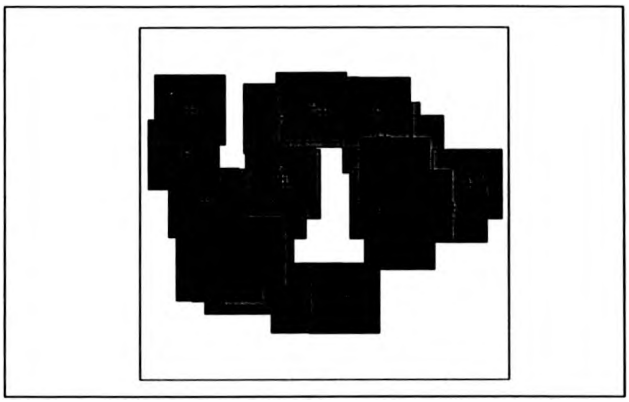

Figure 5: This is a part of the realisation of the attractive interaction in Figure 4. The locations of the incidents are marked by crosses and dots. Let y denote the incidents marked as crosses and z the incidents marked as dots. Then the shaded area is $y \oplus K$, the darker shaded squares show $z \oplus K$. Notice that $z \oplus K \subseteq y \oplus K$ and thus according to Theorem 3.5 the points z are a realisation of a Poisson process on $(y \oplus K) \ominus K$.

The set $(y \oplus K) \ominus K$ is called the *closing* of y by the grain K. In the following we condition the process X to satisfy

$$y \quad \subseteq \quad X \quad \subseteq \quad (y \oplus K) \ominus K.$$

Then X is of the form $X = y \cup Z$ with $Z \subset (y \oplus K) \ominus K$. Let $q(z)$ denote the density of $Z = X \backslash y$ on the closing of y by K. Suppose the point u is in $(y \oplus K) \ominus K$, but not in y. Then we can compute the Papangelou conditional intensity $\lambda_q(z; u)$ of the conditional process Z as follows:

$$\lambda_q(z; u) \quad = \quad \frac{q(z \cup \{u\})}{q(z)} \quad = \quad \frac{p(z \cup y \cup \{u\})}{p(z \cup y)} \quad = \quad \beta,$$

where p is the area-interaction density. Now, let $\alpha(y)$ be the normalizing constant of the density of a Poisson process of intensity β with respect to a unit rate Poisson process on $(y \oplus K) \ominus K$. We now show by induction with respect to number of points $n(z)$ that $q(z)$ is the density of a Poisson process of intensity β on $(y \oplus K) \ominus K$. For $n(z) = 0$ we simply set $q(\emptyset) = \alpha(y)$. Now assume that for $n(z) = n - 1$

$$q(z) \quad = \quad \alpha(y)\, \beta^{n(z)} \quad = \quad \alpha(y)\, \beta^{n-1}.$$

Then

$$q(z \cup \{u\}) \quad = \quad q(z)\, \lambda_q(z, u) \quad = \quad \alpha(y)\, \beta^{n-1}\, \beta \quad = \alpha(y) \beta^{n(z \cup \{u\})},$$

which completes the proof of the Theorem. □

3.4.2 Testing Procedure

As shown in Section 3.2, F and G are defined in terms of distances between points. However, since in practice X is observed within a bounded window W only, inter-point distances based on $X \cap W$ for points close to the border may well differ from the 'true' distances. To deal with these *edge effects*, we map the point pattern onto a torus and regard the observation window as the centre of a 9×9 grid of windows with identical point patterns. The incidents in the other windows are taken into account when determining the nearest neighbour distances in the centre window, see the illustration in Figure 6.

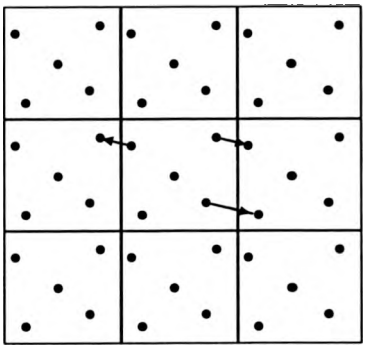

Figure 6: The torus type edge correction: the point pattern is mapped onto a torus and distances to incidents outside of the centre window, like the distances indicated by the arrows, are taken into account.

Note that the torus condition is only one of many possible ways to deal with edge effects [119]. In general the torus condition performs well, although it sometimes looses power for regular point patterns [116]. Surprisingly, tests based on the J-function do not become less powerful when they use naive estimators which are not corrected for edge effects, see [4].

We estimate the summary statistics F, G and J as follows. For each instance we determine the distance to its nearest neighbour and use the empirical cumulative distribution function of these distances as an estimator for G. The estimation of F requires a set of sampling points, which can be a regular grid or a random set of points. We followed the recommendation in Diggle [33] and used a square lattice of sampling points with mesh size

0.1. For each of these sampling points the distance to the nearest instance is determined and the empirical distribution function of these distances yields an estimator for F. Finally, the J-function is simply estimated by a ratio estimate.

If the observed pattern is compatible to the null hypothesis of complete spatial randomness then the estimated summary function $\hat{H}_1$ of the observed pattern should be close to the theoretical Poisson summary function H_0. Recall that under the hypothesis of complete spatial randomness the empty space function F_0 and the nearest neighbour distance distribution G_0 coincide and are given by

$$F_0(r) \quad = \quad G_0(r) \quad = \quad 1 - \exp\left(-\lambda\pi r^2\right).$$

The J-function under complete spatial randomness is equal to $J_0(r) = 1$.

No adequate approximations to the sampling distribution of any of the three summary functions are known, so in order to assess whether the estimated summary function provides sufficient evidence against the null hypothesis we need to resort to a *Monte Carlo test* [10]. Monte Carlo tests were first considered by Barnard in 1963 [6] and are based on the following reasoning. The value of a test statistic u_1 for the data is compared with values $u_2, \ldots, u_m$ of the same statistic obtained from $m-1$ independent simulations of the null hypothesis. The rank of u_1 then yields an exact test, since under the null hypothesis $\mathbf{P}(u_1 = u_{(j)}) = \frac{1}{m}$, where $u_{(j)}$, $j \in \{1, \ldots, m\}$, denotes the

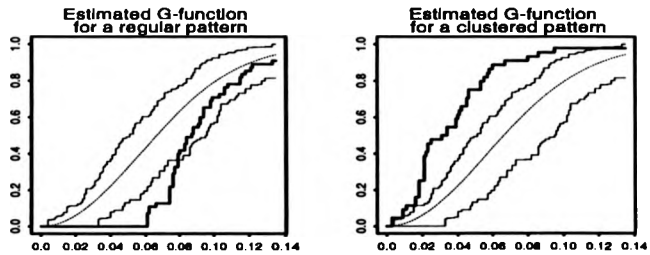

Figure 7: The empirical G-function of the regular and clustered pattern in Figure 3 is shown as a thick solid line. For comparison the Poisson G-function (dotted line) and the 5%-envelopes of 99 Poisson point samples (thin solid lines) are added. Note that for the regular pattern the estimated G-function lies below the Poisson G-function; vice versa for the clustered pattern.

---

j^{th} order statistic.

A graphical Monte Carlo test can be performed as follows, see also the illustration in Figures 7 to 9. A plot of the estimate $\hat{H}_1$ is compared to the upper 5%-envelope U and the lower 5%-envelope L of the estimated summary functions from $m-1$ independent Poisson point samples. Note that these Poisson point samples should be based on the same intensity as assumed for the observed pattern. If the estimated summary function $\hat{H}_1$ of the observed pattern lies outside of these envelopes, then this can be seen as evidence against the null-hypothesis as for each $r \geq 0$:

$$\mathbb{P}\big(U(r) \leq \hat{H}_1(r)\big) \quad = \quad \mathbb{P}\big(L(r) \geq \hat{H}_1(r)\big) \quad = \quad \frac{5}{m}.$$

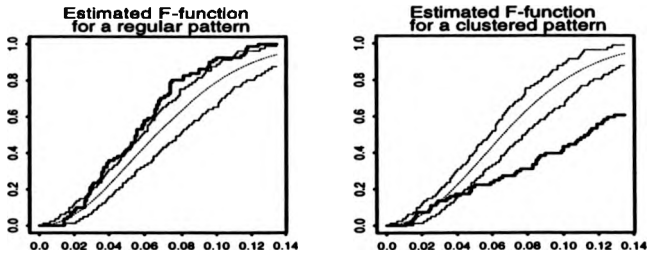

Figure 8: The empirical F-function of the regular and clustered pattern in Figure 3. The dotted line is the Poisson F-function, the thin solid lines are the 5%-envelopes. Notice that for the regular pattern the estimated F-function lies above the Poisson F-function, but for the clustered pattern it lies below the Poisson F-function.

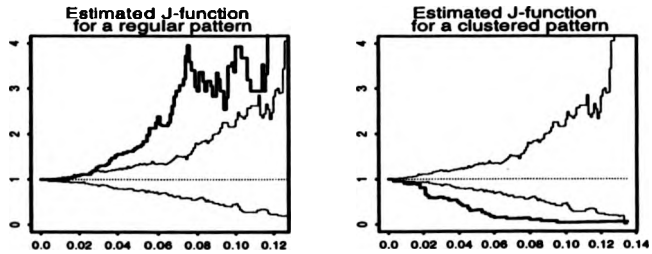

Figure 9: The estimated J-function of the regular and the clustered pattern in Figure 3 (thick line) with the theoretical Poisson J-function (dotted line) and the 5%-envelopes (thin lines). The estimated J-function of the regular pattern is greater than one and of the clustered pattern less than one.

For a more formal test we need to define a test statistic based on a measure of discrepancy between the estimated and theoretical null hypothesis values of a given summary function. If some prior knowledge about the process to be tested is available, for example its interaction radius, then it might be sufficient to use a statistic which is based on the difference between the theoretical and estimated summary function at only one specific value of r:

$$u_1(r) = |\hat{H}_1(r) - H_0(r)|.$$

However, it is usually more sensible to compare values over a range of r. More specifically, we consider the following two test statistics.

1. **Maximum Statistic**

$$u_1 = \max_{0 \le r \le S} |\hat{H}_1(r) - H_0(r)|$$

2. **Integral Statistic**

$$u_1 = \int_0^S (\hat{H}_1(r) - H_0(r))^2 dr$$

Here S denotes the upper limit to the range of r-values considered.

3.5 Results

In the simulation study below, the null hypothesis was tested against the three alternative models discussed in Section 3.4. We varied the parameter

settings and thus the degree of interaction for each model with an overall intensity of 50 points per unit area. For each parameter setting we simulated 100 point patterns and performed the Monte Carlo tests. The number of rejected patterns serves as an estimate for the power of the test.

Monte Carlo tests were performed using the Maximum and Integral Statistics on a significance level of 5%. As Diggle [33] points out, it is usually sufficient to base the test on $m - 1 = 99$ independent samples of the Poisson process since for greater m the power of the test increases only marginally with m. Thus 99 realisations of a Poisson point process with intensity $\lambda = 50$ were generated and the null hypothesis was rejected if the rank of u_1 was 96 or above. Regarding the range of r-values, recall that $J(r)$ is only defined for r such that $F(r) < 1$. Simulations suggest that for our sampling scheme the probability that the estimate of F becomes less than 1 is sufficiently low for values of r up to 0.12 and we therefore estimated the summary functions up to range 0.12.

3.5.1 Cluster Process

First consider the Matérn cluster processes with parent intensity $50/\mu$. The degree of interaction depends on the mean number μ of daughter points and the radius R of the disk on which all daughters with common parent lie. The more daughter points or the smaller the radius R, the more clustered the

point pattern will be. This is reflected in the results summarised in Tables 1 – 4, where the estimated power of all tests decreases with increasing radius R or decreasing mean cluster size μ. We found that tests based on J were the most powerful if based on the Integral Statistic. The least powerful alternative was based on F, see Tables 1 – 4. This stands in contrast to Diggle's results [33], who estimated the power of tests based on F and G using the Maximum statistic. He found that for clustered patterns tests based on F were more powerful than tests based on G. Diggle's study differs from our set-up in a variety of aspects. He fixed the number of points to 100 in each pattern which reduces the variance of his estimates. More importantly, he did not consider a Matérn cluster process but a Thomas cluster field, which has a weaker interaction than the former cluster process. Furthermore, he computed the Maximum statistic for only one fixed upper limit S. But for large values of S the power of G is often considerably reduced, see Tables 1 – 8. In order to see whether the choice of location distribution has an influence on whether tests based on F or G are more powerful, we decided to estimate the power of tests for realisations of the Thomas cluster process. Like Diggle we found that for patterns with weak clustering tests based on F are more powerful than tests based on G at large values of S. However, for stronger clustered patterns tests based on G are most powerful. The effect is more pronounced when reducing the location standard deviation σ then when increasing the cluster size μ. It

is not surprising that tests based on G turn out to be more powerful than tests based on F for patterns with a very small location standard deviation. If the cluster members lie very close to each other then the distances from a sampling point to an incident of a cluster will be approximately the same for all members of the same cluster. Thus the estimated F-function will be close to the F-function of a Poisson process with intensity $50/\mu$. How about the G-function? Most minimum inter-incident distances will correspond to the distance to the nearest member of the same cluster, which is strongly affected by changes in the location standard deviation. Hence G will deviate strongly from the G-function under complete spatial randomness and we expect this deviation to be stronger than the corresponding deviation of F. For the Thomas process the J-function has a performance similar to that of the more powerful alternative among F and G. For $\mu = 5$ it is even the most powerful test statistic for most values of the upper bound S.

For all cluster process alternatives the power of tests based on J reduces considerably for large values of S. This is due to the fact that the variance of the estimator for $J(r)$ increases drastically for large values of r because the estimated $F(r)$ approaches 1. This is more noticable for the Maximum Statistic than for the Integral Statistic as the latter is more robust and hence less affected by random fluctuations.

	Maximum			Integral		
S	F	G	J	F	G	J
0.01	0.09	0.32	0.25	0.08	0.29	0.22
0.02	0.02	0.49	0.42	0.06	0.43	0.37
0.03	0.08	0.64	0.58	0.05	0.60	0.54
0.04	0.11	0.70	0.76	0.08	0.64	0.67
0.05	0.20	0.68	0.73	0.18	0.67	0.75
0.06	0.28	0.66	0.72	0.23	0.72	0.75
0.07	0.39	0.70	0.63	0.29	0.71	0.81
0.08	0.44	0.68	0.34	0.38	0.70	0.78
0.09	0.48	0.65	0.14	0.44	0.69	0.71
0.10	0.51	0.65	0.00	0.48	0.69	0.56
0.11	0.52	0.64	0.00	0.51	0.66	0.29
0.12	0.53	0.62	0.00	0.52	0.65	0.05

Table 1: Estimated power for a Matérn cluster process with $\mu = 2$, $R = 0.08$.

	Maximum			Integral		
S	F	G	J	F	G	J
0.01	0.07	0.15	0.16	0.06	0.17	0.17
0.02	0.03	0.34	0.22	0.03	0.30	0.21
0.03	0.10	0.40	0.35	0.10	0.39	0.34
0.04	0.14	0.53	0.49	0.14	0.45	0.49
0.05	0.26	0.51	0.50	0.19	0.53	0.55
0.06	0.30	0.49	0.47	0.28	0.56	0.58
0.07	0.35	0.44	0.28	0.29	0.51	0.56
0.08	0.36	0.42	0.14	0.37	0.50	0.58
0.09	0.39	0.38	0.05	0.39	0.49	0.44
0.10	0.44	0.38	0.00	0.42	0.45	0.26
0.11	0.46	0.38	0.00	0.44	0.46	0.07
0.12	0.46	0.38	0.00	0.46	0.44	0.01

Table 2: Estimated power for a Matérn cluster process with $\mu = 2$, $R = 0.1$.

	Maximum			Integral		
S	F	G	J	F	G	J
0.01	0.03	0.64	0.64	0.03	0.56	0.52
0.02	0.02	0.91	0.91	0.04	0.87	0.85
0.03	0.09	0.99	0.97	0.09	0.96	0.98
0.04	0.22	1.00	1.00	0.15	0.98	0.98
0.05	0.53	0.99	1.00	0.36	1.00	1.00
0.06	0.67	0.97	1.00	0.56	0.99	1.00
0.07	0.73	0.97	0.98	0.69	0.99	1.00
0.08	0.79	0.97	0.88	0.74	0.99	1.00
0.09	0.82	0.97	0.51	0.79	0.97	1.00
0.10	0.85	0.97	0.02	0.81	0.97	1.00
0.11	0.84	0.97	0.00	0.82	0.97	0.87
0.12	0.84	0.97	0.00	0.83	0.96	0.36

Table 3: Estimated power for a Matérn cluster process with $\mu = 5$, $R = 0.08$.

	Maximum			Integral		
S	F	G	J	F	G	J
0.01	0.08	0.44	0.40	0.09	0.41	0.36
0.02	0.13	0.76	0.66	0.11	0.69	0.63
0.03	0.16	0.90	0.86	0.14	0.86	0.83
0.04	0.30	0.89	0.93	0.24	0.91	0.92
0.05	0.43	0.95	0.98	0.34	0.94	0.96
0.06	0.50	0.93	0.99	0.48	0.96	0.99
0.07	0.66	0.91	0.96	0.55	0.95	1.00
0.08	0.70	0.92	0.77	0.63	0.95	1.00
0.09	0.75	0.92	0.33	0.71	0.95	1.00
0.10	0.78	0.92	0.01	0.73	0.93	0.97
0.11	0.79	0.92	0.00	0.75	0.93	0.76
0.12	0.79	0.92	0.00	0.78	0.92	0.17

Table 4: Estimated power for a Matérn cluster process with $\mu = 5$, $R = 0.1$.

	Maximum			Integral		
S	F	G	J	F	G	J
0.01	0.05	0.14	0.13	0.06	0.13	0.12
0.02	0.05	0.18	0.11	0.04	0.15	0.10
0.03	0.07	0.20	0.13	0.07	0.18	0.14
0.04	0.06	0.31	0.24	0.08	0.27	0.16
0.05	0.11	0.29	0.22	0.08	0.30	0.22
0.06	0.13	0.32	0.24	0.15	0.31	0.28
0.07	0.21	0.27	0.14	0.17	0.35	0.31
0.08	0.22	0.27	0.12	0.18	0.35	0.25
0.09	0.25	0.23	0.03	0.21	0.30	0.18
0.10	0.27	0.22	0.00	0.26	0.28	0.13
0.11	0.31	0.22	0.00	0.28	0.28	0.05
0.12	0.32	0.22	0.00	0.31	0.28	0.00

Table 5: Estimated power for a Thomas cluster process with $\mu = 2$, $\sigma = 0.06$.

	Maximum			Integral		
S	F	G	J	F	G	J
0.01	0.04	0.26	0.21	0.07	0.28	0.21
0.02	0.10	0.49	0.40	0.08	0.40	0.41
0.03	0.07	0.73	0.64	0.08	0.65	0.61
0.04	0.10	0.77	0.68	0.08	0.76	0.73
0.05	0.16	0.75	0.74	0.11	0.79	0.80
0.06	0.20	0.75	0.73	0.20	0.76	0.84
0.07	0.31	0.69	0.62	0.22	0.78	0.81
0.08	0.36	0.67	0.40	0.26	0.77	0.80
0.09	0.41	0.64	0.11	0.36	0.73	0.72
0.10	0.43	0.63	0.01	0.37	0.73	0.63
0.11	0.44	0.63	0.00	0.42	0.73	0.30
0.12	0.45	0.63	0.00	0.45	0.73	0.01

Table 6: Estimated power for a Thomas cluster process with $\mu = 2$, $\sigma = 0.04$.

	Maximum			Integral		
S	F	G	J	F	G	J
0.01	0.04	0.33	0.29	0.03	0.25	0.25
0.02	0.05	0.55	0.47	0.05	0.47	0.45
0.03	0.06	0.68	0.64	0.09	0.63	0.60
0.04	0.23	0.77	0.76	0.13	0.73	0.76
0.05	0.31	0.76	0.81	0.25	0.77	0.81
0.06	0.45	0.73	0.81	0.37	0.80	0.84
0.07	0.55	0.76	0.84	0.45	0.81	0.91
0.08	0.60	0.76	0.57	0.55	0.84	0.93
0.09	0.64	0.74	0.27	0.59	0.80	0.91
0.10	0.71	0.74	0.01	0.62	0.79	0.80
0.11	0.74	0.74	0.00	0.66	0.78	0.51
0.12	0.74	0.74	0.00	0.71	0.77	0.08

Table 7: Estimated power for a Thomas cluster process with $\mu = 5$, $\sigma = 0.06$.

	Maximum			Integral		
S	F	G	J	F	G	J
0.01	0.03	0.66	0.58	0.08	0.62	0.55
0.02	0.01	0.91	0.87	0.03	0.82	0.82
0.03	0.09	0.98	0.99	0.08	0.96	0.96
0.04	0.30	0.99	0.98	0.18	0.99	0.99
0.05	0.49	0.99	0.99	0.37	0.99	1.00
0.06	0.61	0.99	1.00	0.50	0.99	1.00
0.07	0.73	0.98	1.00	0.67	1.00	1.00
0.08	0.75	0.98	0.90	0.71	1.00	1.00
0.09	0.80	0.98	0.48	0.73	0.99	1.00
0.10	0.83	0.98	0.02	0.77	0.99	1.00
0.11	0.84	0.98	0.00	0.80	0.99	0.90
0.12	0.85	0.98	0.00	0.83	0.97	0.33

Table 8: Estimated power for a Thomas cluster process with $\mu = 5$, $\sigma = 0.04$.

3.5.2 Hard Core Process

For the hard core model the strength of interaction increases with the hard core distance. The results for this process are summarised in Tables 9 – 10. It can be seen that when h increases, the power of all tests considered increases as well. For the hard core process, generally tests based on the F-function are least powerful. This is consistent with Diggle's results [33] who investigated the power of tests based on F and G for a sequential inhibition process which is a type of hard core process conditioned on the number of points.

Tests based on G and J are similar in power; the J-function is slightly more powerful for stronger repulsive interaction, in particular if the Integral statistic is used. As in the case of cluster process alternatives, some decrease in power is noticable for J-based tests when S gets large, but the effect is much less pronounced. As mentioned before the loss of power is due to the increasing variance of the estimate of J both for the tested pattern and for the Poisson point patterns. The variance for the estimate of J is bounded for clustered processes as the estimate will take values between 0 and 1. In contrast there is no such bound for Poisson point patterns or hard core patterns, see also Figure 9. Thus due to averaging effect the decrease in power when testing hard core patterns is smaller than when testing cluster processes.

	Maximum			Integral		
S	F	G	J	F	G	J
0.01	0.05	0.00	0.01	0.04	0.00	0.00
0.02	0.04	0.00	0.04	0.02	0.00	0.04
0.03	0.03	0.06	0.30	0.04	0.00	0.17
0.04	0.05	0.83	0.80	0.03	0.64	0.57
0.05	0.04	0.81	0.69	0.05	0.81	0.72
0.06	0.05	0.60	0.50	0.05	0.65	0.68
0.07	0.02	0.40	0.36	0.03	0.49	0.54
0.08	0.02	0.34	0.30	0.03	0.42	0.49
0.09	0.01	0.31	0.24	0.02	0.33	0.35
0.10	0.01	0.30	0.17	0.03	0.29	0.28
0.11	0.01	0.29	0.16	0.03	0.27	0.25
0.12	0.01	0.29	0.10	0.02	0.24	0.19

Table 9: Estimated power for a hard core process with $h = 0.04$.

	Maximum			Integral		
S	F	G	J	F	G	J
0.01	0.08	0.00	0.03	0.09	0.00	0.00
0.02	0.04	0.00	0.06	0.05	0.00	0.00
0.03	0.03	0.06	0.27	0.05	0.00	0.20
0.04	0.07	0.80	0.79	0.05	0.61	0.59
0.05	0.14	0.89	0.99	0.08	0.88	0.96
0.06	0.13	0.90	1.00	0.11	0.90	1.00
0.07	0.19	0.92	1.00	0.14	0.90	1.00
0.08	0.21	0.91	0.96	0.15	0.90	0.99
0.09	0.19	0.91	0.89	0.19	0.90	0.97
0.10	0.19	0.91	0.74	0.17	0.91	0.92
0.11	0.19	0.91	0.67	0.16	0.91	0.87
0.12	0.19	0.91	0.42	0.13	0.91	0.72

Table 10: Estimated power for a hard core process with $h = 0.06$.

3.5.3 Area-interaction Process

The two parameters γ and k, the sidelength of the square K, influence the strength of the interaction in an area-interaction process. If g denotes the logarithm of γ, then for $g < 0$ the model exhibits repulsion, whereas $g > 0$ will lead to aggregation. The density of the area-interaction process weighs a Poisson process density according to an exponential of the area of $x \oplus K$, which is a functional of the empty space. Thus it is surprising that tests based on the empty space function F are the least powerful, both in the repulsive as in the attractive case. However if we increase (respectively decrease) the area of $x \oplus K$, the distance between the points of x will increase (decrease) overproportionally, which provides the intuition why tests based on G are more powerful for area-interaction processes. The power of tests based on J is similar to that of tests based on G, see Tables 11 – 14. We chose not to vary the sidelength k, as the influence of the sidelength on the strength of interaction is ambigous, see also the comments in Section 3.4.1. From Tables 11 and 12, in the repulsive case the power of tests based on J is slightly better than that of tests based on G, in particular if used with the Integral Statistic. In the attractive case the power of tests based on J is similar to the power of tests based on G, see Tables 13 and 14. However, as for cluster processes, the power of J-based tests reduces considerably if the upper limit S of the Integral and Maximum Statistics is increased.

	Maximum			Integral		
S	F	G	J	F	G	J
0.01	0.03	0.00	0.00	0.04	0.01	0.00
0.02	0.04	0.00	0.01	0.03	0.00	0.00
0.03	0.06	0.03	0.07	0.06	0.00	0.03
0.04	0.07	0.24	0.24	0.06	0.20	0.12
0.05	0.06	0.38	0.38	0.06	0.31	0.29
0.06	0.04	0.40	0.53	0.07	0.43	0.50
0.07	0.09	0.50	0.59	0.03	0.51	0.66
0.08	0.08	0.52	0.61	0.05	0.57	0.65
0.09	0.06	0.51	0.59	0.06	0.55	0.69
0.10	0.06	0.50	0.56	0.07	0.55	0.66
0.11	0.06	0.50	0.47	0.07	0.53	0.61
0.12	0.06	0.50	0.32	0.06	0.50	0.49

Table 11: Estimated power for a repulsive area-interaction process with $g = -200, k = 0.1$.

	Maximum			Integral		
S	F	G	J	F	G	J
0.01	0.02	0.00	0.01	0.03	0.00	0.01
0.02	0.05	0.00	0.03	0.05	0.00	0.02
0.03	0.04	0.03	0.12	0.04	0.00	0.05
0.04	0.06	0.47	0.25	0.05	0.37	0.21
0.05	0.06	0.59	0.59	0.04	0.57	0.41
0.06	0.06	0.74	0.68	0.03	0.72	0.66
0.07	0.10	0.78	0.75	0.04	0.80	0.78
0.08	0.11	0.79	0.77	0.04	0.81	0.82
0.09	0.07	0.78	0.76	0.06	0.81	0.82
0.10	0.05	0.77	0.71	0.06	0.82	0.80
0.11	0.05	0.77	0.59	0.06	0.82	0.77
0.12	0.05	0.76	0.45	0.04	0.81	0.67

Table 12: Estimated power for a repulsive area-interaction process with $g = -300, k = 0.1$.

	Maximum			Integral		
S	F	G	J	F	G	J
0.01	0.04	0.28	0.27	0.04	0.28	0.24
0.02	0.04	0.46	0.38	0.05	0.42	0.33
0.03	0.05	0.65	0.53	0.06	0.60	0.52
0.04	0.04	0.66	0.56	0.05	0.66	0.61
0.05	0.04	0.61	0.53	0.06	0.69	0.64
0.06	0.13	0.58	0.53	0.06	0.66	0.68
0.07	0.20	0.56	0.34	0.12	0.65	0.67
0.08	0.26	0.54	0.15	0.18	0.60	0.56
0.09	0.24	0.46	0.03	0.20	0.57	0.40
0.10	0.26	0.46	0.00	0.26	0.53	0.28
0.11	0.29	0.46	0.00	0.28	0.51	0.07
0.12	0.30	0.46	0.01	0.30	0.50	0.02

Table 13: Estimated power for an attractive area-interaction process with $g = 180, k = 0.1$.

	Maximum			Integral		
S	F	G	J	F	G	J
0.01	0.05	0.38	0.37	0.08	0.39	0.32
0.02	0.07	0.70	0.66	0.07	0.67	0.60
0.03	0.08	0.82	0.78	0.09	0.78	0.72
0.04	0.15	0.87	0.91	0.10	0.86	0.84
0.05	0.19	0.90	0.90	0.15	0.90	0.90
0.06	0.25	0.92	0.91	0.16	0.93	0.95
0.07	0.36	0.88	0.82	0.28	0.91	0.94
0.08	0.44	0.87	0.58	0.34	0.92	0.94
0.09	0.47	0.85	0.17	0.41	0.87	0.89
0.10	0.51	0.85	0.00	0.44	0.85	0.74
0.11	0.54	0.85	0.00	0.49	0.84	0.43
0.12	0.56	0.84	0.00	0.55	0.83	0.09

Table 14: Estimated power for an attractive area-interaction process with $g = 250, k = 0.1$.

3.6 Summary and Conclusions

It was hoped that combining F and G into one statistic would yield more powerful tests for complete spatial randomness. Previously, Diggle [33] proposed to use a statistic based on the difference of F and G, but tests based on this statistic had a smaller power than the more powerful among the tests based on G and the tests based on F. Our simulation study examined the statistic J which is also based on a comparison between F and G. We found that using the J-function to test for complete spatial randomness is a competitive alternative to G and F. The J-function produces tests whose power is similar to that of the more powerful of the alternatives F and G. For repulsive processes tests based on J are often slightly more powerful than tests based on the alternative summary functions. However, the performance of J-based tests considerably worsens as the range of values taken into account grows, especially when using the Maximum Statistic. This is due to the fact that the J-function is a ratio and the variance of its estimator increases drastically as the denominator approaches zero. The loss of power is more pronounced for processes with attractive interaction. Therefore we would recommend to consider only those ranges for which the empty space function is sufficiently below 1 and furthermore, to use the Integral Statistic, which is less affected by sampling fluctuations.

4 Markov Chain Theory and Markov Chain Monte Carlo

This chapter provides some theoretical background for the subsequent chapters on perfect simulation. The first section focusses on basic definitions in Markov chain theory which will be used in the following chapters. All the simulation algorithms in the subsequent chapters use coupling methods, thus we provide a short introduction into the theory of couplings in Section 4.2. The following Section 4.3 then gives a brief overview on Markov chain Monte Carlo methods and introduces the idea of perfect simulation which will be further discussed and extended in the subsequent chapters of this thesis.

4.1 Markov Chain Theory

In this section we give a brief overview of the concepts in Markov chain theory used in the following chapters. A similar overview can be found in [144]; for a more thorough discussion including all proofs see [90, 105].

We assume that $(E, \mathcal{E})$ is a measurable space with separable σ-algebra $\mathcal{E}$. Let $P(\cdot, \cdot)$ be the transition kernel of the Markov chain X on the state space E, that is

$$P(x, A) = \mathbf{P}\left(X_{n+1} \in A \,\middle|\, X_n = x\right) \quad \text{for all } x \in E \text{ and } A \in \mathcal{E}.$$

We assume that X has the *invariant distribution* π, that is

$$\pi P(A) = \int P(x, A)\pi(dx) = \pi(A), \qquad A \in \mathcal{E}. \tag{14}$$

Thus, if the chain X is started in the invariant distribution π then it will have distribution π at all subsequent time-points. A sufficient condition for π to be the invariant distribution of X is that π preserves *detailed balance*:

$$\int_B P(x, A)\pi(dx) = \int_A P(x, B)\pi(dx) \qquad \text{for all } A, B \in \mathcal{E}. \tag{15}$$

If P is a transition matrix, then we define the *time-reversal* of P as

$$\tilde{P}(x, y) = \frac{\pi(y)\, P(y, x)}{\pi(x)} \qquad \text{if } \pi(x) > 0. \tag{16}$$

If P and π have densities p and f with respect to a σ-finite measure μ, then we define the time-reversal of the transition kernel density p as

$$\tilde{p}(x, y) = \frac{f(y)\, p(y, x)}{f(x)} \qquad \text{if } f(x) > 0. \tag{17}$$

If the invariant distribution of X satisfies equation (15) then X is time-reversible. The time-reversal kernel $\tilde{P}$ is then identical to P.

Definition 4.1 (φ-irreducibility) *Let φ be a σ-finite, non-trivial measure on $(E, \mathcal{E})$. We call X* φ-irreducible *if for all $x \in E$ and all $A \in \mathcal{E}$ with $\varphi(A) > 0$ there is some positive integer n, possibly depending on x and A, such that $P^n(x, A) > 0$.*

In other words, if X is φ-irreducible then any φ-positive set A will by reached by the chain in finite time with positive probability, regardless in which state x the chain starts.

Definition 4.2 (Aperiodicity) *The φ-irreducible chain X is called* aperiodic *if there are no disjoint sets $A_1, \ldots, A_d \in \mathcal{E}$ such that for $i \in \{1, \ldots, d\}$ and $x \in A_i$ we have $P(x, A_j) = 1$ where $j = i + 1 \mod d$.*

If X is aperiodic then there are no subsets of the state space which the chain can only visit at certain regularly spaced times.

Recall for the following definitions that π denotes the invariant distribution of X.

Definition 4.3 (Positive Recurrence) *We call the π-irreducible, aperiodic chain X* positive recurrent *if for all $A \in \mathcal{E}$ with $\pi(A) > 0$*

$$P\Big(X_n \in A \text{ infinitely often } \Big| X_0 = x \Big) > 0 \quad \text{for all } x \in E \text{ and} \tag{18}$$

$$P\Big(X_n \in A \text{ infinitely often } \Big| X_0 = x \Big) = 1 \quad \text{for } \pi - \text{almost all } x \in E. \tag{19}$$

A stronger form of recurrence for which we do not have to consider null-sets is *Harris recurrence.*

Definition 4.4 (Positive Harris recurrence) *The π-irreducible, aperiodic chain X is* positive Harris recurrent *if for all $A \in \mathcal{E}$ with $\pi(A) > 0$*

$$P\Big(X_n \in A \text{ infinitely often } \Big| X_0 = x \Big) = 1 \quad \text{for all } x \in E. \tag{20}$$

We call the chain X with invariant distribution π *ergodic* if it is π-irreducible, aperiodic and positive Harris recurrent. The distribution of an ergodic chain converges towards its invariant distribution π, that is for all $x \in E$ we have

$$||P^n(x,\cdot) - \pi|| \;=\; 2\sup_{A\in\mathcal{E}}|P^n(x,A) - \pi(A)| \;\longrightarrow 0 \;\text{ as } n \to \infty, \qquad (21)$$

where $||\cdot||$ denotes the variation norm. For a proof of this convergence result see [90, Chapter 13]. If the chain X with invariant distribution π is ergodic then we call π the *equilibrium distribution* or *stationary distribution* of X.

The following definition concerns the rate of convergence.

Definition 4.5 (Geometric and uniform ergodicity) *An ergodic Markov chain X is called* geometrically ergodic *if there exist a non-negative function M such that $E_\pi(M(X)) < \infty$ and a positive constant $0 \leq r < 1$ such that*

$$||P^n(x,\cdot) - \pi|| \quad \leq \quad M(x)\, r^n$$

for all $x \in E$ and $n \in \mathbf{N}$.

The chain is uniformly ergodic *if there is a finite constant m such that $M(x) \leq m$ for all $x \in E$.*

Here is a condition [90, Theorem 16.0.2] which implies uniform ergodicity.

Proposition 4.6 *The Markov chain X is uniformly ergodic, if its state space E is small, that is if there exists a positive integer m and a non-trivial measure ν_m such that for all $x \in E$ and $A \in \mathcal{E}$*

$$P^m(x, A) \quad \geq \quad \nu_m(A).$$

Markov chain Monte Carlo methods estimate characteristics of a distribution π by sampling an ergodic Markov chain with invariant distribution π. The following law of large numbers [90, Theorem 17.1.17] justifies this procedure. Suppose f is a real-valued π-integrable function. Then for any initial distribution of X the sample averages $\frac{1}{N}\sum_{n=1}^{N} f(X_n)$ of the ergodic Markov chain X can be used as estimates for $\mathbf{E}_\pi(f(X))$ due to the following law of large numbers:

$$\frac{1}{N}\sum_{n=1}^{N} f(X_n) \quad \longrightarrow \quad \mathbf{E}_\pi\big(f(X)\big) \qquad \text{almost surely as } n \to \infty. \tag{22}$$

More details on Markov chain Monte Carlo methods can be found in Section 4.3. But first we give a short introduction into the theory of coupling. This theory is not only helpful when examining properties of Markov chains but is also the basis of the new perfect Markov chain Monte Carlo techniques.

4.2 Coupling Theory

It is generally agreed that the first use of coupling dates back to Doeblin's paper [36] from 1938. But only in the 1970's, due to the growing interest in

interacting particle systems, did coupling become an established method for stochastic processes; see the historical remarks in [84].

Doeblin [36] used coupling to show that a finite, irreducible and aperiodic Markov chain converges to a statistical equilibrium. Since this first use of couplings, the method has had a strong impact on the asymptotic theory of Markov chains. A recent approach which uses couplings to assess through simulation the convergence of a Markov chain can be found in Johnson [65]. Furthermore, the coupling idea has turned out to be essential in the development of the exact Markov chain Monte Carlo algorithms which will be presented in the subsequent chapters. This section will provide a short discussion on coupling, focussing on concepts which are used for perfect simulation algorithms.

Coupling is a comparison of probability measures on a measure space. Essentially it defines a joint distribution for given marginals. A more accessible definition is given in [84] in terms of random elements. Suppose $(E, \mathcal{E})$ is a measurable space, $(\Omega, \mathcal{F}, \mathbb{P})$ a probability space and X a measurable mapping from Ω to E.

Definition 4.7 (Coupling) *A coupling of the random elements* $(\Omega, \mathcal{F}, \mathbb{P}, X)$ *and* $(\Omega', \mathcal{F}', \mathbb{P}', X')$ *in* $(E, \mathcal{E})$ *is a random element* $(\hat{\Omega}, \hat{\mathcal{F}}, \hat{\mathbb{P}}, (\hat{X}, \hat{X}'))$ *in the product space* $(E^2, \mathcal{E}^2)$ *such that*

$$\mathcal{L}(X) = \mathcal{L}(\hat{X}) \qquad \textit{and} \qquad \mathcal{L}(X') = \mathcal{L}(\hat{X}').$$

Thus a coupling is a bivariate random element $(\hat{X}, \hat{X}')$ with given marginals.

The couplings considered in this thesis will be of a more restrictive nature: two stochastic processes are coupled if their paths coincide after a random time T, the *coupling time*. Here is a more formal definition.

Definition 4.8 (Coupling time) *If $(\hat{X}, \hat{X}')$ is a coupling of the processes $X = \{X_n, n \geq 0\}$ and $X' = \{X'_n, n \geq 0\}$ in $(E, \mathcal{E})$, then the random time T such that $\hat{X}_n = \hat{X}'_n$ for $n \geq T$ is a* coupling time. *If $T < \infty$ almost surely we call the coupling* successful.

Example 4.9 *As a simple example consider Doeblin's asymptotic stationarity result, which is summarised in [142] as follows. Suppose P is an irreducible and aperiodic Markov transition matrix. Let X and X' be two Markov chains with the same transition matrix P but different initial values $X_0 = i$ and $X'_0 = k$. The chains are run independently until they meet, say at random time T. From then onwards let X and X' run together. Due to irreducibility and aperiodicity there exist some finite integer m and a constant $\epsilon > 0$ such that $P^m(l, j) > \epsilon$ for all l, j. It follows that $\mathbb{P}(T > km) \leq (1 - \epsilon)^k$ and thus $\mathbb{P}(T > km) \to 0$ as $k \to \infty$. As a result the chains will eventually meet and hence*

$$|\mathbb{P}(X_n = j) - \mathbb{P}(X'_n = j)| \leq \mathbb{P}(X_n \neq X'_n) \longrightarrow 0 \quad \text{as } n \to \infty. \qquad (23)$$

Due to the Chapman-Kolmogorov equation, $\max_i P^n(i, j)$ is non-increasing

and $\min_i P^n(i,j)$ *is non-decreasing in* n *and so it follows that* $\mathbb{P}(X_n = j)$ *has a limit.*

This idea can be extended to provide a bound on the variation distance of a Markov chain from stationarity. Suppose $X = \{X_n, n \geq 0\}$ is an ergodic Markov chain with equilibrium distribution π. Assume $X_0 = x$. Let $X' = \{X'_n, n \geq 0\}$ be a Markov chain with the same transition kernel as X but started in equilibrium. Then we obtain the following *coupling inequality*

$$\begin{aligned} ||P^n(x,\cdot) - \pi|| &= ||\mathcal{L}(X_n) - \mathcal{L}(X'_n)|| \\ &= 2 \sup_{A \in \mathcal{E}} |\mathbb{P}(X_n \in A) - \mathbb{P}(X'_n \in A)| \\ &= 2 \sup_{A \in \mathcal{E}} |\mathbb{P}(X_n \in A, X_n \neq X'_n) - \mathbb{P}(X'_n \in A, X_n \neq X'_n)| \\ &\leq 2\,\mathbb{P}(X_n \neq X'_n) = 2\,\mathbb{P}(T > n), \end{aligned} \tag{24}$$

where T is the coupling time for X and X'.

For Markov chains coupling of transitions can be arranged using so-called *transition* or *update rules.* These are deterministic functions together with a random variate and are usually an explicit part of the simulation algorithm.

Definition 4.10 (Transition rule) *A* transition rule *for the transition kernel* P *is a measurable function* $f : E \times S \to E$ *together with a random variable* U *on some state space* S *with*

$$\mathcal{L}\big(f(x,U)\big) = P(x,\cdot) \qquad \textit{for all } x \in E.$$

Suppose we have a partially ordered state space $(E, \leq)$. *Then we call the transition rule* $f(\cdot, U)$ monotone *if it is monotone in its first argument:*

$$f(x,u) \quad \geq \quad f(y,u) \quad \textit{whenever } x \geq y \quad \textit{for } x, y \in (E, \leq) \textit{ and } u \in S.$$

Similarly, we call $f(\cdot, U)$ anti-monotone *if*

$$f(x,u) \quad \leq \quad f(y,u) \quad \textit{whenever } x \geq y \quad \textit{for } x, y \in (E, \leq) \textit{ and } u \in S.$$

Although a transition rule is only completely specified by both the function f *and* the random variable U, in the following we will often not refer to U but assume that the distribution of U is fixed.

Suppose f is a transition rule for P. Let $X = \{X_n, n \geq 0\}$ and $X' = \{X'_n, n \geq 0\}$ be two Markov chains with the same transition kernel P, but with different starting values x_0 and x'_0 respectively. Then a coupling $(\hat{X}, \hat{X}')$ of X and X' is defined by setting $(\hat{X}_0, \hat{X}'_0) = (x_0, x'_0)$ and

$$\left(\hat{X}_n, \hat{X}'_n\right) \quad = \quad \left(f(\hat{X}_{n-1}, U),\ f(\hat{X}'_{n-1}, U)\right).$$

Naturally the properties of a coupling of a Markov chain are closely linked to the properties of its transition kernel. The majority of perfect simulation algorithms will rely on some monotonicity property of the transition kernel. In the following we assume that E is a Polish space which is endowed with a partial order $\leq$.

Definition 4.11 (Monotone transition kernel) *Let P be a transition kernel on $(E, \leq)$. Then P is* monotone *if for all $A(z) = \{w \in E : w \geq z\}$, $z \in E$, we have*

$$P(x, A(z)) \quad \geq \quad P(y, A(z)) \qquad \text{whenever } x \geq y,$$

that is if $P(x, \cdot)$ stochastically dominates $P(y, \cdot)$ whenever $x \geq y$.

We call P anti-monotone *if for all $A(z)$, $z \in E$,*

$$P(x, A(z)) \quad \leq \quad P(y, A(z)) \qquad \text{whenever } x \geq y.$$

The following proposition [67, Theorem 1], which is a corollary to Strassen's theorem [135], shows that we can use couplings to prove monotonicity of a transition kernel. It uses the notion of an *upward kernel* which is a transition kernel $K(\cdot, \cdot)$ such that $K(x, A(x)) = 1$ for all $x \in E$.

Proposition 4.12 *Let X and X' be two Markov chains with transition kernel P and initial states $X_0 = x$ and $X'_0 = x'$ where $x \leq x'$. Then the following statements are equivalent:*

1. *The transition kernel P is monotone.*
2. *There exists a coupling $(\hat{X}, \hat{X}')$ of X and X' such that*

$$\hat{X}_n \quad \leq \quad \hat{X}'_n \quad \text{for } n \in \mathbf{N}.$$

3. There exists an upward kernel $K_{x,x'}$ on $(E, \mathcal{E})$ such that

$$P(x', A) = \int K_{x,x'}(y, A) P(x, dy).$$

Assume P is monotone, then the second statement in Proposition 4.12 means that we can simultaneously produce a sample path of X and a sample path of X' such that the sample path of X will always lie below the path of X'. The third statement means that given a sample path of X we can produce a sample path of X' such that this path always lies above the path of X.

If the transition kernel P has a monotone transition rule, then P is monotone. Note that the reverse is not true. Fill and Machida [39] call a transition kernel which has a monotone transition rule *realizably monotone* and furthermore present an example of a monotone transition kernel which is not realizably monotone.

Given a monotone transition rule for P, we can define a monotone transition rule for the upward kernels of P satisfying the third condition in Proposition 4.12 as follows.

Remark 4.13 *Let $f(\cdot, U)$ be a monotone transition rule for P. Assume that for $x, x' \in E$ the conditional distribution of U given $f(x, U) = x'$ is well-defined and that the random variable $\hat{U}$ has this distribution. Then, for $y \geq x$, the transition rule $f(y, \hat{U})$ is distributed according to the upward*

kernel $K_{x,y}(x', \cdot)$ *of* P *with*

$$P(y, A) \quad = \quad \int K_{x,y}(x', A) P(x, dx').$$

Thus we can use $f(y, \hat{U})$ *to sample from this upward kernel.*

The subsequent chapters of this thesis will illustrate how couplings and, more particularly, transition rules can be used to produce exact samples of an equilibrium distribution.

4.3 Markov Chain Monte Carlo

This chapter provides a compact overview on Markov chain Monte Carlo algorithms; more detailed overviews can be found in [14, 42, 48, 143].

4.3.1 Introduction and History

A probabilistic model which is sufficiently realistic and flexible often leads to a distribution over a high-dimensional space. Examples for such complex distributions include point process densities in stochastic geometry, Markov random fields in statistical physics and posterior distributions in Bayesian statistics. In the following assume that the distribution of interest has a density $\pi(\cdot)$ with respect to a σ-finite measure $\mu(\cdot)$. To ease notation we use π to denote both the target distribution and its density.

Many distributions on high dimensional spaces are analytically intractable and thus a characterisation is only possible by means of statistical analysis. Suppose f is some measurable function on the state space of π. *Monte Carlo integration* draws independent samples $X_1, \ldots, X_N$ from $\pi(\cdot)$ and then uses sample averages $\frac{1}{N}\sum_{n=1}^{N} f(X_n)$ to estimate the expectation $E_\pi[f(X)]$.

Often direct sampling of the density $\pi(\cdot)$ is not feasible as its normalising constant is not known in closed form and the numerical evaluation of the constant is not possible. *Markov chain Monte Carlo* (MCMC) methods base the sampling of π on a Markov chain. An ergodic Markov chain with equilibrium distribution π is sampled after it has run for a long time. Due to the convergence result (21) the samples will be approximately (or even exactly) distributed according to π. A law of large numbers (22) then ensures that the sample averages $\frac{1}{N}\sum_{n=1}^{N} f(X_n)$ approximate the expectation $E_\pi[f(X)]$.

The first MCMC algorithm [88] can be found in the statistical physics literature and since then many of the earliest and most important methods have been developed in statistical physics and image analysis. Metropolis et al. (1953) [88] were the first to consider sampling based on Markov chains; their *Metropolis algorithm* produces (approximate) samples of a Gibbsian point process. Hastings (1970) [57] generalised the algorithm to the more flexible *Metropolis-Hastings algorithm*, which is applicable to a wide variety of statistical problems. Besides the Metropolis-Hastings algorithm the

most popular MCMC method is the *Gibbs Sampler*, which was developed by Geman and Geman [45] in 1984 for applications in image restoration. A special case of the Gibbs Sampler for Markov random fields, the *heat bath algorithm*, was already known to the statistical physics literature in the 1970's, see for example [26]. Probably the first description of the heat bath transition kernel can be found in the 1963 paper by Glauber [49]. The name "heat bath" derives from the fact that sites, when interacting with an external heat reservoir, tend to swap their spin. Gelfand and Smith (1990) [43] extended Gibbs Sampling to continuous distributions and illustrated its applicability to Bayesian inference problems.

Recently auxiliary variable techniques have been developed which introduce additional variables to the Markov chain in order to improve its performance. Auxiliary variables might improve the speed of convergence or simplify the simulation by avoiding the sampling from non-standard distributions. The idea of auxiliary variables was first used by Swendsen and Wang [138] to produce a faster MCMC algorithm for the simulation of the Ising and the Potts model. Further algorithms using auxiliary variables can be found in [58, 92, 101, 122].

Within stochastic geometry, *spatial birth-and-death processes* provide an alternative MCMC method. Following the introduction of spatial birth-and-death processes by Preston [110] in 1975, Ripley [115, 116] formulated the

so-called *add-delete algorithm* for the simulation of finite point processes. Geyer and Møller [46] developed in 1994 a Metropolis-Hastings algorithm for finite point processes. A generalization of this method, which is also based on Markov jump processes, is the *reversible jump Markov chain Monte Carlo* which was developed by Green (1995) [50] for problems such as Bayesian model determination.

In the past three years a new variant of Markov chain Monte Carlo methods has been developed, the perfect simulation algorithms. These are algorithms which check whether the simulated Markov chain has reached equilibrium yet and thus are able to produce samples which are guaranteed to be from the stationary distribution.

The next two sections present the most common MCMC methods: the Metropolis-Hastings algorithm and the Gibbs Sampler. We then discuss some MCMC techniques specifically for point processes. In the subsequent section we point out some implementational issues and the final section introduces the new idea of perfect simulation.

4.3.2 Gibbs Sampler

The Gibbs Sampler is based on conditioning: we split the Markov chain $X = (X_1, \ldots, X_d)$ into d components and update each component in turn according to the full conditional distribution given by the densities $\pi(x_k \mid x_{-k})$

where $x_{-k} = \{x_j, j \neq k\}$. The Gibbs Sampler is particularly convenient if the full conditional distributions are some standard distributions which are easy to sample. For example, the parametrization of X could be chosen to make use of a conditional independence structure between components.

The order in which we update the components can be either random or sequential. For sequential updating the resulting transition kernel is not reversible, although the individual components are. The transition kernel for sequential updating is given by

$$P(x, A) \quad = \quad \int_A \prod_{i=1}^{d} \pi(y_i \,|y_{j<i}, x_{j>i})\, \mu(dy),$$

where $y_{j<i} = (y_1, \ldots, y_{i-1})$ and $x_{j>i} = (x_{i+1}, \ldots, x_d)$. Conditions for the ergodicity of a Gibbs chain can be found in [121, 123].

4.3.3 Metropolis-Hastings Algorithm

The Metropolis-Hastings algorithm [57] is based on a kind of rejection sampling: transitions of the Markov chain are proposed according to a Markov transition kernel and then accepted according to a probability which ensures that the resulting Markov chain has invariant distribution π.

Let $q(x, y)$ be the density with respect to $\mu(\cdot)$ of a Markov transition kernel, this is the proposal density. Suppose now the Markov chain is currently in state $X_n = x$. A candidate state y for X_{n+1} is sampled from $q(x, \cdot)$. A transition of the chain to y is then accepted with probability $\alpha(x, y)$ and the

chain is set to $X_{n+1} = y$. However, if the transition is rejected the chain stays in its current state, that is $X_{n+1} = x$. The acceptance probability is given by the Metroplis-Hastings ratio

$$\alpha(x, y) \quad = \quad \min\Big\{1, \frac{\pi(y)q(y, x)}{\pi(x)q(x, y)}\Big\}. \tag{25}$$

The transition kernel of the Metropolis-Hastings chain is given by

$$P(x, A) \quad = \quad \int_A q(x, y)\alpha(x, y)\mu(dy) + r(x)1_{[x \in A]}, \tag{26}$$

where $r(x) = 1 - \int q(x, y)\alpha(x, y)\mu(dy)$ is the rejection probability. Notice that the construction of this chain only depends on π through the ratio $\pi(y)/\pi(x)$ and thus we do not need to know the normalising constant of the density $\pi(\cdot)$. Since $q(x, y)\alpha(x, y)$ satisfies the detailed balance equation (15) with respect to π it follows that π is the invariant distribution of the Metropolis-Hastings chain. To ensure that the chain converges to π the proposal density needs to be chosen such that the resulting chain is ergodic. Some simple conditions for the ergodicity of a Metropolis-Hastings chain can be found in [123].

The original Metropolis algorithm [88] assumes that the proposal density is symmetric, that is $q(x, y) = q(y, x)$. The acceptance probability (25) then simplifies to

$$\alpha(x, y) \quad = \quad \min\Big\{1, \frac{\pi(y)}{\pi(x)}\Big\}.$$

4.3.4 Markov Chain Monte Carlo for Finite Point Processes

This section describes methods which are designed to sample a finite point process which has a density f with respect to a homogeneous unit rate Poisson process. Note that we use the same notation as in Section 2.2.2. We present two alternative methods: spatial birth-and-death processes and Metropolis-Hastings algorithms.

Spatial birth-and-death processes are spatio-temporal point processes. At any given time the spatial birth-and-death process forms a point pattern in $\mathbb{R}^d$. This pattern changes at distinct time instances. The change can be either a *birth*, that is a point is added to the current pattern, or a *death*, that is a point is deleted from the current point configuration. Since these changes depend only on the current point pattern, spatial birth-and-death processes are continuous-time Markov jump processes. Spatial birth-and-death processes can be characterised through a birth and a death rate. The birth rate is a measurable function $b : \mathbb{R}^d \times \Omega \to [0, \infty)$ such that $\int_B b(\xi, x) d\xi < \infty$, where B is a bounded Borel set on the state space Ω, the family of finite point patterns on $\mathbb{R}^d$. Given the current configuration x at time t, the probability of a birth in B during the short time interval $[t, t + s)$ is given by $s \int_B b(\xi, x) d\xi + o(s)$.

The death rate is a measurable function $d : \mathbb{R}^d \times \Omega \to [0, \infty)$. Given the current configuration $x \cup \{\xi\}$ at time t, the probability that ξ is deleted

during a time interval $[t, t+s)$ is given by $s\, d(\xi, x)+o(s)$. Conditions on b and d which ensure the existence and ergodicity of the spatial birth-and-death process can be found in [110]. The rate of convergence of ergodic spatial birth-and-death processes is examined in [85, 94].

A spatial birth-and-death process whose invariant distribution has density f is time-reversible if the detailed balance condition

$$f(x)\, b(\xi, x) \quad = \quad f(x \cup \{\xi\})\, d(\xi, x) \tag{27}$$

is satisfied, where $f(x \cup \{\xi\})$ is assumed to be positive. If the invariant distribution of the process describes a finite Markov point process on a window W, then the birth death process is time-reversible if $b(\xi, x) = 1$ and $d(\xi, x) = \lambda(x, \xi)^{-1}$, where $\lambda(\cdot, \cdot)$ denotes the Papangelou conditional intensity. For a unit birth rate the location of a newly born point is uniform over W. Although the generation of new points is easy for a unit birth rate, Ripley [115] observed that it might lead to slow convergence as newly born points are frequently removed shortly after birth. He proposed to choose instead a constant death rate $d(\xi, x) = 1$ and birth rate $b(\xi, x) = \lambda(x, \xi)$. In this case new points can be generated using rejection sampling. Having decided on the birth and death rate the spatial birth-and-death process is simulated as follows. An initial point configuration is chosen, this could be for example a realisation of a Poisson point process. Assume that at time t the process had a birth or a death and the resulting configuration is equal

to $x = (x_1, \ldots, x_k)$. The time to the next incident is an exponential random variable of rate

$$k + \int_W \lambda(x, \xi) d\xi.$$

Then with probability

$$\frac{1}{k + \int_W \lambda(x, \xi) d\xi}$$

the next incident is the death of point x_j for some $j \in \{1, \ldots k\}$. Otherwise we have the birth of a point whose location ξ is distributed according to the density

$$\frac{\lambda(x, \xi)}{\int_W \lambda(x, \eta) d\eta}.$$

Often in spatial statistics point processes are conditioned to have a fixed number of points, as the number of points is regarded as an ancillary statistic. Ripley [115] developed an MCMC variant which produces Markov point patterns conditional on the number of points. The method is based on a discrete-time Markov chain in which births and deaths alternate. The birth rate $b(\xi, x)$ and death rate $d(\xi, x)$ is zero for any pattern x which does not have $n-1$ points. If x has $n-1$ points then we may choose $b(\xi, x) = \lambda(x, \xi)$ and set the death rate $d(\xi, x)$ to 1. Alternatively we may set the birth rate $b(\xi, x)$ constant to 1. Then the death rate has to be $d(\xi, x) = \lambda(x, \xi)^{-1}$. As births and deaths alternate this algorithm has become known as the add-delete algorithm.

Recently, Geyer and Møller [46] developed an alternative to the spatial birth-and-death process approach: a Metropolis-Hastings algorithm which samples a point process with a random number of points. Consider the state space $E = \{f > 0\}$. We follow the presentation in [95] which only considers births and deaths as transitions. This is a special case of the algorithm in [46] which also allows the replacement of a point as a transition. Let $X_n = x$ be the current point configuration. With probability $p(x)$ the birth of a point ξ, which is sampled according to a density $b(x, \cdot)$, is proposed; the birth is accepted with a probability $\min\{1, \alpha(x, \xi)\}$. Alternatively the death of a point $\eta \in x$, which is chosen randomly with probability $d(x \backslash \eta, \eta)$, is proposed and its death is accepted with probability $\min\{1,\ \alpha(x \backslash \eta, \eta)^{-1}\}$. The acceptance probabilities are specified by the Metropolis-Hastings ratio

$$\alpha(x, \xi) \quad = \quad \lambda(x, \xi)\, \frac{1 - p(x)}{p(x)}\, \frac{d(x, \xi)}{b(x, \xi)}. \tag{28}$$

To ensure that the constructed chain is time-reversible the following conditions need to hold

$$p(x) > 0, \qquad p(x \cup \{\xi\}) < 1, \qquad d(x, \xi) > 0, \qquad b(x, \xi) > 0,$$

where $x \cup \{\xi\} \in E$. If furthermore $p(\emptyset) < 1$ then the chain is aperiodic, π-irreducible, where π is the distribution of the point process with density f, and positive Harris recurrent. Green [50] developed a more general framework, known as reversible jump MCMC, which encompasses the algorithm

developed by Geyer and Møller [46].

A Metropolis-Hastings algorithm to sample a point process with fixed number of points is also available and described in [95].

4.3.5 Implementational Issues

The above algorithms construct Markov chains based on one transition kernel, however it is also possible to combine transition kernels with the same invariant distribution.

All statistical inference from MCMC samples is based on the assumption that the constructed Markov chain converges to a statistical equilibrium. This assumption leads to a variety of implementational issues such as the choice of starting points, the length of burn-in periods and the number of runs. Although the distribution of the Markov chain will eventually become independent of the starting value, the choice of starting points can influence the speed of convergence. Furthermore, to reduce the influence of the starting value a number of initial iterates, the so-called *burn-in*, is usually discarded. A difficult but essential problem is to determine which starting values to choose and how long the burn-in should be to limit the influence of the starting values. Thus it is not surprising that much MCMC research has focussed on the computation or estimation of convergence rates. Although some theoretical results are available, see for example [46, 121, 124, 127],

they are usually not generally applicable. Moreover, they might not provide explicit bounds or, if bounds can be computed, they might be too weak to be of any practical use. Some information about convergence can be derived from convergence diagnostics which assess whether the chain has converged either a posteriori or during the runtime; a review on convergence diagnostics can be found in [15, 22]. But, although these diagnostics might warn if convergence has not yet been reached, they cannot guarantee that the chain is sufficiently close to equilibrium.

A further issue is whether the inference should be based on one very long run of the chain or several shorter parallel runs. Proponents of the single long run, see for example [47], argue that as the chain has run for a longer time it is more likely to be close to equilibrium. Furthermore, several shorter runs are wasteful as for each run an initial burn-in has to be discarded. On the other hand proponents for several short runs, see for example [44], argue that the use of several shorter runs may guard against the possibility that the chain has not explored essential parts of the sample space. Also, if several chains are started in different states then some information about convergence can be derived by comparing the output.

4.3.6 Perfect Simulation Algorithms

The previous Section 4.3.5 has illustrated that the question of convergence, although essential for any inference based on MCMC, poses difficult problems. A remedy to most problems would be provided by an algorithm which is able to verify itself whether the chain has reached stationarity and thus produces exact rather than approximate samples from the equilibrium distribution. Such algorithms have recently become available and are known as perfect simulation methods.

These algorithms can assess, during run time, whether the Markov chain has yet reached equilibrium and thus produce samples which are *exactly* distributed according to the target distribution.

The term *perfect* rather than *exact* simulation was first used by Kendall [73] and is motivated in [74, page 2]: "For several reasons simulations cannot be 'exact' in the precise sense: useable random number generators always have defects (even if not yet discovered!), while ... one should admit into one's analysis the possibility that the algorithm fails to deliver an answer within practical constraints of time". Thus in a "perfect" environment without defects in random generators, machine rounding errors and time constraints perfect simulation algorithms would produce exact samples.

At present there are two types of perfect simulation algorithms. The first type is based on the *Coupling from the Past* idea which was developed

by Propp and Wilson in 1995 [111]. This method has received most of the attention and the vast majority of articles on perfect simulation deal with applications or generalisations of Coupling from the Past. Chapter 5 will present the Coupling from Past method and give a literature review. The following Chapter 6 will discuss an extension of the technique to a random set model.

An alternative perfect simulation method was proposed by Fill in 1997 [37] and has become known as the *interruptible algorithm*. This method is discussed in Chapter 7 which also presents an extension of the method to a point process example.

5 Coupling from the Past

5.1 The Essence of Coupling from the Past

Throughout the following sections we assume that X is an ergodic Markov chain with stationary distribution π. Suppose X is started at time 0 in some initial distribution ρ_0. Then, as time t tends to infinity, the distribution of X at time t converges to the stationary distribution, that is

$$\mathcal{L}(X_t^0) \quad = \quad \rho_t \quad \longrightarrow \quad \pi \quad \text{as} \quad t \longrightarrow \infty.$$

Sampling X at time $t \geq 0$ produces a sample from ρ_t. If there was a way of continuing sampling for an infinite amount of time, we would be able to sample from

$$\lim_{t \to \infty} \rho_t \quad = \quad \pi.$$

As this is not feasible, conventional Markov chain Monte Carlo samples ρ_S for some large but *finite* S and relies on the assumption that ρ_S is sufficiently close to π. As discussed in Section 4.3.5 it is difficult to assess how large S should be such that ρ_S is a sufficiently good approximation to π.

Coupling from the Past (CFTP) is based on a different approach, which is also illustrated in Figure 10. Suppose we were able to start X at time $-\infty$ in some initial distribution $\pi_{-\infty}$ and run it up to time 0. Then, heuristically we may say that the distribution of X converges to the stationary distribution as t approaches zero, that is for $t \leq 0$

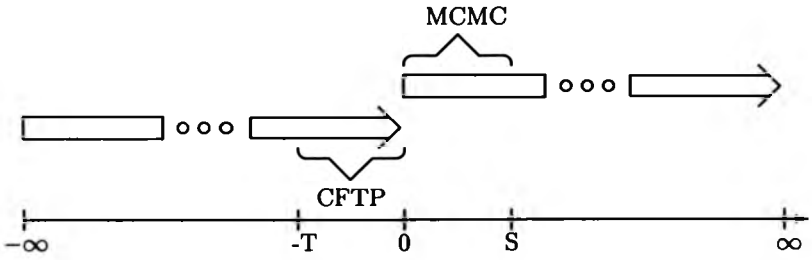

Figure 10: Whereas MCMC samples the initial part of an infinite time simulation, CFTP samples the final part.

$$\pi_t \quad = \quad \mathcal{L}(X_t^{-\infty}) \quad \longrightarrow \quad \pi \quad \text{as} \quad t \longrightarrow 0,$$

and so

$$\pi_0 \quad = \quad \pi. \tag{29}$$

CFTP makes use of this heuristic by reconstructing a simulation from time $-\infty$ for a finite time interval in the recent past. This reconstruction delivers the state of $X^{-\infty}$ at time 0 which according to (29) is a sample from the equilibrium distribution. Thus CFTP is able to produce perfect, that is exact samples from π. The time interval $[-T, 0]$ on which CFTP reconstructs the path of the infinite time simulation is random and dynamically determined by the algorithm.

The essence of all the CFTP algorithms presented in this and the following chapter is to produce such a "virtual simulation from time $-\infty$" by examining

a simulation on a finite time interval in the recent past.

In the next Section 5.2 we discuss the CFTP algorithm in its simplest form, see also Propp and Wilson [111]. Since its introduction, various authors have extended the original CFTP algorithm in a variety of aspects. Section 5.3 is an attempt to describe CFTP in the most general form available to date. This framework enables us in Section 5.4 to classify the CFTP algorithms which currently can be found in the literature. The last Section 5.5 summarizes further results and comments on CFTP.

5.2 Coupling from the Past in its Simplest Form

We shall begin by describing Coupling from the Past in its simplest form, Propp and Wilson [113] call this version of Coupling from the Past *Voter CFTP* because of its relation to the voter model for interacting particle systems. Assume the Markov chain X with transition matrix P lives on a finite state space E. Our aim is to reconstruct an infinite time simulation of X on a finite time interval in the recent past. A main tool is *simulation backwards in time*. What do we mean by simulation backwards in time? Rather than sampling a transition step from a single state, we sample a random map which specifies a transition step for *each* state in the state space, that is a

transition rule. We assume that the chosen transition rule f is such that

$$\mathbf{P}(\, f(x,U) = f(y,U)\,) \quad \geq \quad \sum_{z \in E} P(x,z)P(y,z).$$

Note that equality can always be achieved by the independent coupling.

We go step by step backwards in time and at each step we independently obtain a realisation of the transition rule. Let $f_{-n}(\cdot) = f(\cdot, U_{-n})$ denote the n^{th} sampled random map. We compose the sampled random maps and thus produce the stochastic flow

$$F_0^{-T} \quad = \quad f_{-1} \circ f_{-2} \circ \ldots \circ f_{-T}, \qquad T \in \mathbf{N}, \tag{30}$$

A *stochastic flow* on the state space E is a random process evolving in time, whose values are random maps from E to E. A survey on the use of stochastic flows in Markov chain theory can be found in [32].

The composite map F_0^{-T} specifies for each state the final image of a path of X started in this state at time $-T$ and run up to time 0. Note the order in which we compose the random maps which extends the stochastic flow backwards rather than forwards in time. As pointed out in [32], there is a fundamental difference in the behaviour of a stochastic flow which is extended backwards and a stochastic flow which is extended forwards in time. For some $x \in E$ define a *forward process* $\{X_n, n \in \mathbf{N}\}$ starting from $X_0 = x$ by setting

$$X_n \quad = \quad \Big(f_{-n} \circ f_{-(n-1)} \circ \ldots \circ f_{-1}\Big)(x), \qquad n \in \mathbf{N}.$$

This forward process evolves like the Markov chain X thus converges in distribution to the stationary distribution π. Now consider the *backwards process* $\{\tilde{Y}_n, n \in \mathbf{N}\}$ by setting $\tilde{Y}_0 = x$ and

$$\tilde{Y}_n \quad = \quad \big(f_{-1} \circ f_{-2} \circ \ldots \circ f_{-n}\big)(x) \quad = \quad F_0^{-n}(x), \qquad n \in \mathbf{N}.$$

In Example 5.4 we show that there exists an almost surely finite time $-T_C \leq 0$ such that $F_0^{-T} = F_0^{-T_C}$ for all $-T \leq -T_C$. Therefore, the backward process converges almost surely to a limit. Thus, although $\tilde{Y}_n$ has the same distribution as X_n for each $n \in \mathbf{N}$ the backward process behaves very differently from the forward process. Figure 12 on page 100 illustrates the difference between the forward and the backward process using the Markov chain presented in Example 5.1, see also the comments below Example 5.1.

Now consider an *"infinite time simulation"* of X started at time $-\infty$ and run up to time $-T$. As the chain is ergodic and has been run for an infinite amount of time, heuristics suggest that the chain is in equilibrium at time $-T$. Suppose that $X_{-T}^{-\infty} = x$, then the state x is a sample of π. The transition rules $f_{-n}, n \in \mathbf{N}$, preserve the equilibrium distribution and so it follows that $F_0^{-T}(x)$ is also a sample of π. The composite map F_0^{-T} therefore specifies the final state of a set of paths which by an heuristic argument includes the path of an infinite time simulation.

Note that as the transitions are coupled through the random maps f_{-n}, $n \in \mathbf{N}$, the paths coalesce as they meet. After each sampling step we check

whether the image of the composite map F_0^{-T} contains only one element. If the image is a singleton then we can conclude that all paths have coalesced into a single path. The unique image of F_0^{-T} may then be seen as the state of an infinite time simulation at time 0. Thus we may accept this state as a sample from the equilibrium distribution π. Theorem 5.2 in the next section shows rigorously why the sample has distribution π.

If the image of F_0^{-T} has more than one element then the paths have not yet coalesced into one and we cannot yet specify the state of the infinite time simulation at time 0. We iteratively go further into the past until the composite map has a one-point (random) image, that is until we achieve complete coalescence. The amount of time we have to go back into the past until we achieve complete coalescence is random. For any implementation of CFTP we need to show that this random time is almost surely finite. For Voter CFTP the coalescence time is almost surely finite, see Example 5.4 on page 106 for a proof.

A clear and efficient way to describe algorithms is to use the pseudocode notation from computer science, indicating blocks by indentation. For example,

---

```
if A then
   B
else
   C
D
```

means, *"if condition A holds then do B else do C. After that, do D".*

Suppose we have an algorithm `RandomMap()` which samples the transition rules of P. Then we can describe the CFTP algorithm as follows:

```
VoterCFTP:
   t ← 0
   F_0 ← identity map
   Repeat
      f ← RandomMap()
      F_t ← F_{t+1} ∘ f
      t ← t − 1
      until image of F_t is a singleton
   return image of F_t
```

Here is an example for Voter CFTP.

Example 5.1 (Simple Random Walk) *Suppose X is a random walk on the state space $E = \{0, 1, 2\}$ with transition matrix*

$$P = \begin{pmatrix} p & q & 0 \\ p & 0 & q \\ 0 & p & q \end{pmatrix}$$

where $p \in (0, 1)$ and $q = 1 - p$. Of course, for this Markov chain we can easily compute and sample the stationary distribution π defined by

$$(p^2 + qp + q^2)\,\pi \quad = \quad \left(p^2,\ pq,\ q^2\right).$$

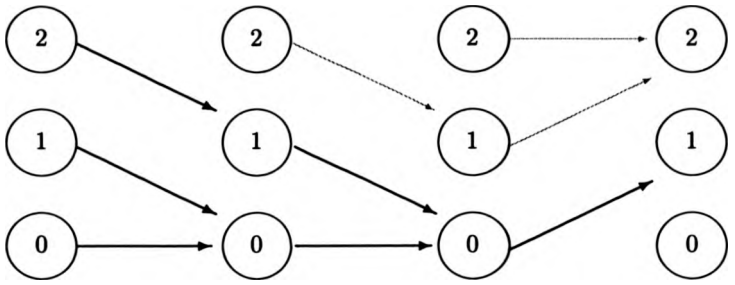

Figure 11: Illustration of Example 5.1. The arrows show the sampled transition rule for each time step. The darker arrows illustrate the composite map F_0^{-3}.

We can simulate X using the transition rule

$$f_{-n}(j) \quad = \quad f(j,\, U_{-n}) \quad = \quad \begin{cases} \max\{j-1,\ 0\} & \text{if } U_{-n} = -1 \\ \min\{j+1,\ 2\} & \text{if } U_{-n} = 1, \end{cases}$$

where $n \in \mathbf{N}$ and $\{U_{-n}, n \in \mathbf{N}\}$ is a sequence of independent Bernoulli random variables taking value 1 with probability p and value -1 with probability $1-p$. By sampling U_{-n} we sample a transition rule f_{-n}. Suppose in the first step we sample $U_{-1} = 1$, then the image of the composite map F_0^{-1} is $\{1,2\}$. As the image is not a singleton we go a step further into the past and independently sample U_{-2}. Say $U_{-2} = -1$ then the image of the composite map F_0^{-2} is $\{1,2\}$ and still contains more than one element. We again go one step backwards and sample $U_{-3} = -1$. Now the composite map F_0^{-3} has the unique image $\{1\}$. Thus we can accept state 1 as a sample from π. Figure 11 illustrates the procedure.

As pointed out before, it is essential that we extend the stochastic flow

F_0^{-T} backwards and not forwards in time. As soon as the image of the flow consist of a singleton, going further backwards in time will not change the (random) image of the stochastic flow. Recall the definition of the forward process X_n and the backwards process $\tilde{Y}_n$ on page 94. Suppose we set $X_0 = \tilde{Y}_0 = 2$ and

$$\begin{aligned} X_n &= \big(f_{-n} \circ f_{-(n-1)} \circ \ldots \circ f_{-1}\big)(2), \\ \tilde{Y}_n &= \big(f_{-1} \circ f_{-2} \circ \ldots \circ f_{-n}\big)(2) = F_0^{-n}(2), \quad n \in \mathbb{N}. \end{aligned}$$

Say we sample $(U_{-1}, \ldots, U_{-6}) = (1, -1, -1, 1, 1, -1)$. Then

$$\begin{aligned} (X_0,\ X_1,\ X_2,\ X_3,\ X_4,\ X_5,\ X_6\ ) &= (2,\ 2,\ 1,\ 0,\ 1,\ 2,\ 1\ ) \\ (\tilde{Y}_0,\ \tilde{Y}_1,\ \tilde{Y}_2,\ \tilde{Y}_3,\ \tilde{Y}_4,\ \tilde{Y}_5,\ \tilde{Y}_6\ ) &= (2,\ 2,\ 1,\ 1,\ 1,\ 1,\ 1\ ), \end{aligned}$$

and so X_n moves through E whereas $\tilde{Y}_n$ converges to the (random) limit 1. Figure 12 illustrates the forward process $\{X_n\}$ and the backward process $\{\tilde{Y}_n\}$ for $n = 1, \ldots 6$.

5.3 Coupling from the Past in its Most General Form

In the following we describe the concept of Coupling from the Past in its greatest generality, including the original algorithm by Propp and Wilson [111] and most extensions known to date.

As before we assume that X is an ergodic Markov chain on E with stationary distribution π. Suppose there is process U on some state space $\mathcal{U}$

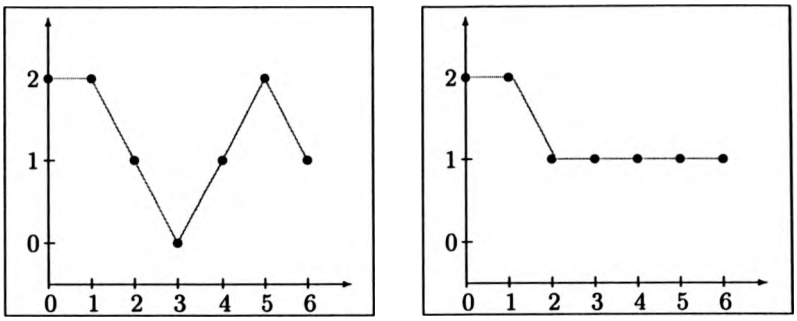

Figure 12: Illustration of the backward and the forward process in example 5.1. The forward process moves through the state space according to the transition probabilities in P and converges in distribution to π. In contrast, the backward process converges to its (random) limit.

---

and a time-homogeneous process $g(U) = \{g(U_{-t}), -t \leq 0\}$ on E which is a functional of U, such that X can be realized as an adapted functional G of U, that is

$$X_{-s}^{-T}\big(g(U_{-T})\big) \quad = \quad G\Big(\{U_{-t}, -T \leq -t \leq -s\};\ g(U_{-T})\Big). \tag{31}$$

Here $\{X_{-s}^{-T}(g(U_{-T})), -s \geq -T\}$ behaves as the chain X started at time $-T$ in the (random) state $g(U_{-T})$. The functional $G(\{U_{-t}, -T \leq -t \leq -s\};\ \cdot)$ usually describes a (coalescing) stochastic flow. The *underlying* process U has to be such that we can extend a realisation of $\{U_{-t},\ -t \in [-T, 0]\}$ to a realisation of $\{U_{-t},\ -t \in [-T - S, 0]\}$ for $-S < 0$.

Suppose furthermore there exists a *bounding* process Y on $\mathcal{P}(E')$, the subsets of some state space $E' \supseteq E$, which is also coupled to U through a functional H, that is

$$Y_{-s}^{-T} \quad = \quad H\Big(\{U_{-t}, -T \leq -t \leq -s\}\Big). \tag{32}$$

Assume that this bounding process satisfies the following requirements: firstly it needs to possess the *funneling* property:

$$Y_{-s}^{-T-u} \subseteq Y_{-s}^{-T} \qquad \text{for all } -u \leq 0 \text{ and } -T \leq -s \leq 0. \tag{33}$$

Thus the earlier the starting time $-T$ of the set-valued process Y^{-T} the smaller the set it will be in at a fixed time $-s \geq -T$.

Secondly, we assume the bounding process has the following *coalescence* properties:

$$\text{if } Y_{-s}^{-T} \text{ is a singleton, then } Y_{-u}^{-T} \text{ is a singleton for } -s \leq -u \leq 0 \tag{34}$$

and

$$T_C = \inf\{T \geq 0 : Y_0^{-T} \text{ is a singleton } \} < \infty \text{ almost surely.} \tag{35}$$

Thus, if started early enough, the bounding process will become a singleton and once it becomes a singleton it will remain a singleton.

Finally, we require the bounding process to have the following *sandwiching* relation with X

$$X_{-s}^{-T}\big(g(U_{-T})\big) \in Y_{-s}^{-T} \qquad \text{for all } -T \leq 0 \text{ and } -T \leq -s \leq 0. \tag{36}$$

Informally, the function of the underlying process U is to provide all the randomness describing the infinite time simulation of X. The bounding process on the other hand is an attempt to reconstruct such an infinite time

simulation. Notice that as a consequence of (33) we have

$$X_{-s}^{-T-u}\big(g(U_{-T-u})\big) \quad \in \quad Y_{-s}^{-T} \quad \text{for all } -u \leq 0.$$

Thus, for all $-S \leq 0$, the bounding process contains the realisation of X started at time $-S$ in the distant past in the initial state $g(U_{-S})$ and coupled to the underlying process through the functional G. Theorem 5.2 shows that therefore the bounding process (heuristically) can be thought to contain a realisation of an infinite time simulation within a finite time interval in the recent past. If the bounding process becomes a singleton then we can derive the state which the infinite time simulation would take at time 0. The proof of Theorem 5.2 will also formally define the term "infinite time simulation".

Note that although Y is based on the same underlying process as X and contains all relevant realisations of X we do not require that

$$H(\cdot) \quad = \quad \bigcup_{x \in \mathcal{X}} G(\cdot, x) \qquad \text{for some } \mathcal{X} \subseteq E.$$

In fact we even allow Y to contain realisations outside of the state space of X; this might be necessary to ensure an efficient implementation of the sandwiching property. An example where the bounding process contains realisations which are not in the state space of the target chain is presented in the following Chapter 6. However, as a result of the sandwiching property (36) the bounding process Y will evolve like X as soon as it becomes a singleton.

The following Theorem 5.2 shows that the above ingredients are sufficient to produce a sample from the equilibrium distribution π.

Theorem 5.2 *Suppose the ergodic Markov chain X can be realized as an adapted functional of some underlying process U as in (31). Furthermore assume there is a bounding process Y which is coupled to the underlying process U as in (32) and satisfies the funneling requirement (33), the coalescence conditions (34) - (35) and has the sandwiching property (36). Set*

$$T_C \quad = \quad \inf\{\, T \geq 0 : \; Y_0^{-T} \textit{ is a singleton } \}, \tag{37}$$

then $Y_0^{-T_C}$ has the equilibrium distribution π.

Proof: The coalescence property (35) states that the coalescence time T_C is almost surely finite and thus the limit $\lim_{T\to\infty} Y_0^{-T}$ exists. As a result of the sandwiching property (36) we also have that

$$\lim_{T\to\infty} Y_0^{-T} \quad = \quad \lim_{T\to\infty} X_0^{-T}\big(g(U_{-T})\big) \quad = \quad X_0^{-\infty}. \tag{38}$$

Thus the limit $X_0^{-\infty}$ represent a *"virtual simulation from time $-\infty$"*. Note that as $g(U)$ is time-homogeneous together with the fact X is an ergodic Markov chain with stationary distribution π we have

$$\mathcal{L}\big(X_0^{-T}(\, g(U_{-T})\,)\big) \quad = \quad \mathcal{L}\big(X_T^0(\, g(U_0)\,)\big) \quad \longrightarrow \pi \quad \text{as } T \longrightarrow \infty \tag{39}$$

Now, (38) shows that $X_0^{-T}\big(g(U_{-T})\big)$ converges to $X_0^{-\infty}$ with respect to the discrete metric which implies weak convergence, see [60]. We conclude that

$$\mathcal{L}\big(X_0^{-\infty}\big) \quad = \quad \pi.$$

Finally note that because of the funneling property (33)

$$X_0^{-\infty} \quad = \quad \lim_{T\to\infty} Y_0^{-T} \quad = \quad Y_0^{-T_C}$$

which completes the proof of Theorem 5.2. □

Remark 5.3 *Theorem 5.2 is a direct extension of Theorem 1 in [74] and the above proof is essentially the same as the proof given therein.*

Theorem 5.2 shows that in order to produce a perfect sample from the stationary distribution of an ergodic Markov chain we need to find an underlying process U as in (31) and a bounding process Y satisfying requirements (33) – (36). A Coupling from the Past algorithm can then be run as follows:

```
GeneralCFTP:
   T ← 0
   u ← ∅
   Repeat
      T ← T + 1
      U ← Extend(u, −T)
      Y[−T, 0] ← H(U)
      until Y_0 is a singleton
   return Y_0
```

Here the algorithm `Extend`$(u, -T)$ produces a realisation of U on $[-T, 0]$ by extending the realisation u backwards in time. The bounding process Y is coupled to U through the functional H.

Although this provides us with a perfect sampling algorithm, the practicality of U and Y for an actual simulation depends on a variety of aspects. First of all we need a way of extending U backwards in time, that is an `Extend` algorithm. Often the underlying process is a sequence of independent and identically distributed random variables, for which a backwards extension is straight-forward. Alternatively, consider an underlying process U which is a Markov chain satisfying detailed balance. Then, if the stationary distribution of U is a standard distribution we simply sample U_0 in equilibrium and continue to simulate U forwards in time. Due to detailed balance the chain U is time-reversible so that we can use the time-reversal of a realisation of $\{U_t, t \geq 0\}$ as a realisation of the underlying process $\{U_{-t}, -t \leq 0\}$. Given a realisation of U on $[-T, 0]$, we simulate from time $-T$ further backwards in time and thus extend the given simulation to a simulation over the time interval $[-T - S, 0]$ for some $S \geq 0$.

Besides the underlying process U we need to identify a bounding process Y which satisfies the requirements (33) – (36). This process needs to be such that there is an efficient way of monitoring when Y becomes a singleton. Only if there is an efficient representation of the set-valued process Y is

CFTP feasible. Many CFTP algorithms exploit for example monotonicity properties of the transition kernel of X for this purpose, these algorithms are described in Sections 5.4.1 and 5.4.2. Finally, although the coalescence time T_C is almost surely finite it might be too large in practice to allow for efficient CFTP simulations. Burdzy and Kendall [16] discuss the concept of efficient Markovian couplings and thus the "price of perfection".

Before we provide an overview on the CFTP algorithms available to date, we give two examples which illustrate the use of Theorem 5.2.

Example 5.4 (Voter CFTP) *Consider the Voter CFTP algorithm of the previous section. Let the function $f(\cdot, V)$ together with the random variable V be a transition rule for the transition matrix P of X. Here the underlying process $U = \{V_{-n}, n \in \mathbb{N}\}$ is a sequence of independent copies of V. Let $f_{-n}(\cdot) = f(\cdot, V_{-n})$ denote the n^{th} sampled transition rule and define*

$$F_{-s}^{-T} \quad = \quad f_{-(s+1)} \circ \cdots \circ f_{-(T-1)} \circ f_{-T} \quad \text{for } -T \leq 0 \text{ and } -T \leq -s \leq 0.$$

For some arbitrary state $x \in E$ set $g(V_{-n}) = x$ for all $n \in \mathbb{N}$, which clearly defines a time-homogeneous process. A realisation of the chain X can then be produced by setting

$$X_{-s}^{-T}(x) \quad = \quad F_{-s}^{-T}(x)$$

and thus X can be realized as an adapted functional of U as in (31). A

bounding process is given by

$$Y_{-s}^{-T} \quad = \quad \bigcup_{x \in E} F_{-s}^{-T}(x) \quad = \quad \text{image}\Big(F_{-s}^{-T}\Big).$$

We now show that Y_{-s}^{-T} possesses the required properties (33) - (36). First notice that $X_{-s}^{-T}\Big(g(V_{-T})\Big) = F_{-s}^{-T}(x)$ clearly is in $Y_{-s}^{-T} = \text{image}(F_{-s}^{-T})$, whence the sandwiching property (36) is satisfied. The funneling property (33) is easily verified by observing that

$$F_{-s}^{-T-u} \quad = \quad F_{-s}^{-T} \circ F_{-T}^{-T-u} \quad \textit{for all } -u \leq 0 \textit{ and } -T \leq -s \leq 0.$$

Moreover,

$$F_{-u}^{-T} \quad = \quad F_{-u}^{-s} \circ F_{-s}^{-T} \quad \textit{for } -s \leq -u \leq 0, \tag{40}$$

hence if Y_{-s}^{-T} is a singleton then Y_{-u}^{-T} is a singleton for $-s \leq -u \leq 0$, thus Y satisfies the coalescence condition (34).

Finally, we need to show that the coalescence time T_C is almost surely finite; this is requirement (35). We follow the proof given in [111]. Recall that we chose the transition rule f such that

$$\mathbf{P}(\, f(x,U) = f(y,U)\,) \quad \geq \quad \sum_{z \in E} P(x,z)P(y,z).$$

Let z be an arbitrary state in the state space E. As X is irreducible and aperiodic there is a finite $N > 0$ such that $P^N(y,z) > 0$ for all $y \in E$. Thus there is a constant $\epsilon > 0$ such that for any $-t \leq 0$

$$\mathbf{P}\left(\text{image}(F_{-t}^{-t-N}) \textit{ is a singleton}\right) \quad > \quad \epsilon.$$

Consider the events

$$\left\{\text{image}(F_{-(k-1)N}^{-kN}) \text{ is a singleton } \right\}, \quad k \in \mathbf{N}.$$

These events are independent and all have a probability of at least ϵ. *Thus by Borel-Cantelli*

$$K \quad = \quad \min\left\{k \in \mathbf{N} : \text{ image } (F_{-(k-1)N}^{-kN}) \text{ is a singleton } \right\}$$

is almost surely finite . If the image of $F_{-(K-1)N}^{-KN}$ *contains only one element it follows with (40) that the image of* F_0^{-KN} *also contains only a single element. Thus requirement (35) holds.*

If we use U and Y as defined above, then the CFTP algorithm `GeneralCFTP` reduces to the algorithm `VoterCFTP` presented in Section 5.2. With Theorem 5.2 it follows that `VoterCFTP` produces samples from the target stationary distribution. Note however that if the state space E is very large, it will be very expensive to monitor the bounding process. In the next Section 5.4 we present methods which allow an efficient representation of the bounding process and thus lead to *practical* CFTP algorithms.

The following example illustrates a CFTP algorithm for a continuous-time Markov chain which lives on an infinite state space and thus is an example for which we cannot use the algorithm `VoterCFTP`. The example is taken from [71] but rephrased to show how it fits into the general framework given

in Theorem 5.2. It shows how to sample the equilibrium distribution of a continuous-time birth-and-death process X on the natural numbers. The transition rates of X are given by a birth rate $\lambda(X)$ and a death rate μ. We assume that the birth rate $\lambda(\cdot)$ is a positive and increasing function which is uniformly bounded above by a constant α. First we need to determine an underlying process.

Example 5.5 (Birth-death process) *Consider a birth-and-death process Z with constant birth rate α and constant death rate μ. This process satisfies detailed balance and has a stationary distribution which is Poisson with mean α/μ. As discussed earlier, if started in equilibrium we can easily extend a simulation of such a process backwards in time. To each arrival or departure time $-t$ of Z we attach an independent uniform $[0,1]$ random variable M_{-t} as a mark. This marked birth-and-death process U functions as the underlying process. We can produce a realisation of X as an adapted functional of U as follows. Let Z_{-t-} denote the configuration of Z immediately prior to time $-t$. We start X at time $-T$ in the initial state Z_{-T} and evolve the chain according to the following rules. Changes of X only occur at transition times of Z. If $-t$ is an* arrival time *in Z, that is $Z_{-t} = Z_{-t-} + 1$, we have an arrival in X at time $-t$ if and only if the mark of the arrival time $-t$ satisfies:*

$$M_{-t} \quad \leq \quad \frac{\lambda(X_{-t-})}{\alpha}.$$

If on the other hand $-t$ is a departure time, *that is $Z_{-t} = Z_{-t-} - 1$, and*

$X_{-t-} > 0$ then we have a departure in X at time $-t$ if and only if the mark of the departure time $-t$ satisfies:

$$M_{-t} \quad \leq \quad \frac{X_{-t-}}{Z_{-t-}}.$$

This construction yields a birth-and-death process with birth rate $\lambda(X)$ and death rate μ. Let G denote the functional describing the above construction, that is G is such that

$$X_{-s}^{-T}(g(U_{-T})) \quad = \quad X_{-s}^{-T}(Z_{-T}) \quad = \quad G\Big(\{U_{-t}, -T \leq -t \leq -s\};\ Z_{-T}\Big).$$

Note that as we start Z in equilibrium, the process $g(U) = Z$ is time-homogeneous. Furthermore, note that the monotonicity of the birth rate $\lambda(\cdot)$ implies that G is monotone in the second argument that is whenever $x \leq y$

$$G\Big(\{U_{-t}, -T \leq -t \leq -s\};\ x\Big) \quad \leq \quad G\Big(\{U_{-t}, -T \leq -t \leq -s\};\ y\Big) \quad (41)$$

The choice of the initial state of X suggests to use the following bounding process:

$$Y_{-s}^{-T} \quad = \quad \bigcup_{0 \leq x \leq Z_{-T}} G\Big(\{U_{-t}, -T \leq -t \leq -s\};\ x\Big).$$

As soon as this bounding process becomes a singleton it remains a singleton, thus it satisfies requirement (34). A sufficient condition for Y_0^{-T} to be a singleton is that Z hits zero in the interval $[-T, 0]$. This will occur for an almost surely finite $-T$, thus the bounding process satisfies the coalescence

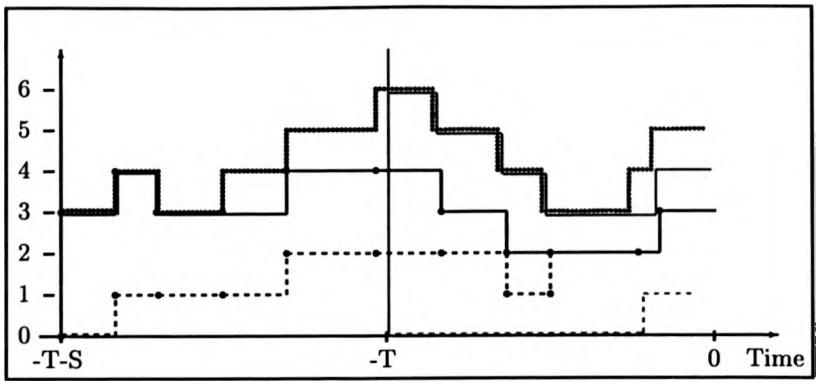

Figure 13: The birth-death process example 5.5: The dotted line is the underlying constant rate birth-death process Z, the dashed line shows the minimal process and the solid line the maximal process. The gray lines show a CFTP run started from time $-T$ in which coalescence of the minimal and maximal process does not occur. The black lines show a run started from time $-T-S$ in which coalescence occurs. Note how the process Z has been extended backwards in time. Note also that the minimal process lies below the maximal process and how both processes satisfy the funneling property.

---

property (35). The funneling property (33) follows from

$$0 \quad \leq \quad G\big(\{U_{-t}, -T \leq -t \leq -s\};\, x\big) \quad \leq \quad Z_{-s} \quad \textit{for all } 0 \leq x \leq Z_{-T}.$$

Finally,

$$X_{-s}^{-T}\big(g(U_{-T})\big) \quad = \quad G\big(\{U_{-t}, -T \leq -t \leq -s\};\, Z_{-T}\big) \quad \in \quad Y_{-s}^{-T}$$

which is the sandwiching property (36). Figure 13 illustrates the relation between Z, X and Y.

Notice that the monotonicity property (41) of G provides us with an easy way of checking whether the bounding process has become a singleton: Y_{-s}^{-T} is a singleton if and only if

$$G\big(\{U_{-t}, -T \leq -t \leq -s\};\, 0\big) \quad = \quad G\big(\{U_{-t}, -T \leq -t \leq -s\};\, Z_{-T}\big).$$

Thus to check whether the bounding process has become a singleton we just need to produce two paths started at time $-T$ and coupled to U through the functional G. One path is started in 0 and the other is started in Z_{-T} and coalescence of the two paths means that the bounding process has coalesced into a singleton.

5.4 Variants of Coupling from the Past - An Overview

This section gives an overview on Coupling from the Past algorithms in the current literature, concentrating on papers which developed main concepts and extensions of the original CFTP algorithm. For a complete collection of research papers related to perfect simulation including CFTP, see the annotated bibliography by Wilson [148]. Most extensions of CFTP provide solutions of how to find an adequate bounding process and how efficiently to monitor it. Finding an efficient represention of the bounding process is essential for the feasibility of any CFTP algorithm.

5.4.1 Monotone CFTP

Consider a partially ordered state space $(E, \preceq)$ with a unique minimal element $\hat{0}$ and unique maximal element $\hat{1}$ such that $\hat{0} \preceq x \preceq \hat{1}$ for all $x \in E$. Let X be an ergodic, discrete-time Markov chain with transition kernel P. Suppose $f(\cdot, V)$ is a *monotone* transition rule for P and $U = \{V_{-n}, n \in \mathbf{N}\}$ is

a sequence of independent copies of V. Then we can use this sequence as an underlying process and, as before, define

$$F_{-s}^{-T} \quad = \quad f_{-(s+1)} \circ \cdots \circ f_{-T}.$$

The stochastic flow F_{-s}^{-T} inherits the monotonicity from the transition rules. Analogous to Example 5.4, we can show that the process

$$Y_{-s}^{-T} \quad = \quad \text{image}(F_{-s}^{-T})$$

is a bounding process satisfying the requirements (33) – (36). Propp and Wilson [111] observed that in this setting a particularly efficient CFTP variant, monotone CFTP, can be constructed. We define a *minimal process*

$$Y_{-s}^{\min(-T)} \quad = \quad F_{-s}^{-T}(\hat{0})$$

and a *maximal process*

$$Y_{-s}^{\max(-T)} \quad = \quad F_{-s}^{-T}(\hat{1}).$$

These two processes sandwich between them the elements of the bounding process, that is

$$Y_{-s}^{-T} \quad = \quad \bigcup \left\{ x \in E : \, Y_{-s}^{\min(-T)} \preceq x \preceq Y_{-s}^{\max(-T)} \right\}$$

Thus Y_{-s}^{-T} is a singleton if and only if

$$Y_{-s}^{\min(-T)} \quad = \quad Y_{-s}^{\max(-T)}.$$

This means we can efficiently monitor whether the bounding process becomes a singleton by simply keeping track of the two paths of the minimal and the maximal process. Notice that both the minimal and the maximal process are Markov chains with transition kernel P and sandwich between them all relevant realisations of X.

Monotone CFTP enabled Propp and Wilson [111] to produce samples of attractive spin systems on very large state spaces. For example they sampled a ferromagnetic Ising model on a 4200×4200 toroidal grid even for a temperature close to criticality. For a review on the Ising model and related Markov random fields see Section 2.2.1. Without exploiting monotonicity, constraints on runtime and computer memory would have made it very difficult to produce a sample on such a large state space.

To produce an exact sample of a ferromagnetic Ising model, Propp and Wilson [111] first produce an exact sample of the random cluster model with parameter $q = 2$ and then assign a common random spin to all the vertices within one connected component. The resulting spin configuration is a sample of the Ising model, see [40].

Say H is a state of the random cluster model on the graph G. Recall that the random cluster model assigns probability $\mathbb{P}(H)$ to the configuration H, where $\mathbb{P}(H)$ is proportional to

$$\Big(\prod_{v,w \in H} p_{v,w} \Big) \Big(\prod_{v,w \notin H} (1 - p_{v,w}) \Big) \, 2^{C(H)}$$

Here $C(H)$ is the number of connected components in H and $p_{v,w}$ is the probability of observing an edge connecting v with w. We use the notation $H\backslash E$ to denote the graph H without the edge E, thus $H\backslash E = H$ if $E \notin H$. Similarly, $H \cup E$ denotes the graph H including the edge E; if $E \in H$, then $H \cup E = H$.

Example 5.6 (Monotone single-bond heat-bath) *The single-bond heat-bath algorithm is a Gibbs Sampler cycling through the set of edges, see [137]. The algorithm chooses an edge E and then decides whether to include the edge according to the conditional probability of observing $H \cup E$ given the remainder $H\backslash E$. The following transition rule describes the Gibbs update of an edge E:*

$$f(H, U) \quad = \quad \begin{cases} H \cup E & \text{if } U \leq \frac{\mathbf{P}(H \cup E)}{\mathbf{P}(H \cup E) + \mathbf{P}(H \backslash E)} \\ H \backslash E & \text{otherwise,} \end{cases}$$

where U is uniform on the unit interval.

Now, note that the subgraph-inclusion defines a partial order on the state space of the random cluster model. The maximal state $\hat{1}$ is given by the complete graph G, the minimal state $\hat{0}$ is the empty graph. It can be show that the transition rule f is monotone with respect to the subgraph-inclusion. Thus the single-bond heat bath algorithm is amenable to monotone CFTP.

We can extend the single-bond heat bath algorithm to a so-called *omnithermal* algorithm, see [111]. An omnithermal algorithm simultaneously

produces a sample for a range of parameter values. The algorithm cycles through the edges and assigns to each edge the set of parameter values for which the edge would be included. If the inclusion of an edge depends monotonically on the value of the model parameter then the omnithermal algorithm can be used within a monotone CFTP algorithm.

An example for monotone CFTP for point processes is presented in Häggström, Van Lieshout and Møller [54].

Example 5.7 (Point Process) *The authors extend monotone CFTP to a point process example on an uncountable state space E. As in the above setting, the transition rule in this example is monotone with respect to a particular partial order. But in contrast to the previous setting the state space does not contain a unique minimal and a unique maximal element with respect to this partial order. However, the authors noticed that there are (non-unique) states x and x' in E such that*

$$Y_{-s}^{-T} \quad = \quad \bigcup \left\{ y \in E : \; F_{-s}^{-T}(x) \preceq y \preceq F_{-s}^{-T}(x') \right\}.$$

In [54] the state x is called quasi-minimal *and the state x'* quasi-maximal. *As a consequence we can define a minimal process $Y_{-s}^{\min(-T)} = F_{-s}^{-T}(x)$ and a maximal process $Y_{-s}^{\min(-T)} = F_{-s}^{-T}(x')$ which can be used in the same way as in Propp and Wilson's setting to monitor whether the bounding process has coalesced into a singleton. (In fact by slightly changing the partial order used*

in [54] we can define states which, although not unique, are maximal and minimal in the usual sense; we comment further on this in Chapter 7. The algorithm in [54] is then monotone CFTP as in [111] with the only difference that maximal and minimal state are not unique.)

Unfortunately the algorithm in [54] is not generally applicable to distributions on infinite state spaces as it is tied to the particular point process example.

Monotone CFTP works analogously for monotone continuous-time Markov chains; here is an example.

Example 5.8 (Birth-Death process) *Recall the CFTP algorithm for a birth-and-death process in Example 5.5. We can define a minimal and a maximal process for a monotone CFTP algorithm as follows. Set*

$$Y_{-s}^{\max(-T)} \quad = \quad G\big(\{U_{-t}, -T \le -t \le -s\};\ Z_{-T}\big)$$

and

$$Y_{-s}^{\min(-T)} \quad = \quad G\big(\{U_{-t}, -T \le -t \le -s\};\ 0\big).$$

This pair of processes satisfies the requirements of a minimal and a maximal process. Note that the maximum process is based on a random maximal element Z_{-T}. We will comment further on the use of random maximal and minimal elements in Section 5.4.3

5.4.2 Anti-Monotone CFTP

Consider again a partially ordered state space $(E, \preceq)$ with unique maximal and minimal element. (We use the same notation as in the previous section.) If the Markov chain is based on *anti-monotone* transition rules, see Definition 4.10, then we can, as before, define a maximal and a minimal process which monitor the bounding process. Set

$$Y_{-T}^{\max(-T)} \;=\; \hat{1} \quad \text{and} \quad Y_{-T}^{\min(-T)} \;=\; \hat{0}$$

and evolve them as follows:

$$Y_{-s}^{\max(-T)} \;=\; f_{-(s+1)}\Big(Y_{-(s+1)}^{\min(-T)}\Big) \quad \text{and} \quad Y_{-s}^{\min(-T)} \;=\; f_{-(s+1)}\Big(Y_{-(s+1)}^{\max(-T)}\Big).$$

Thus a transition in the maximal process is dependent on the current state of the minimal process and vice versa. In the following we call the construction of making transitions of the maximal process dependent on the current state of the minimal process and vice versa a *cross-over*. Minimal and maximal process together evolve as a two-component Markov chain, however they are not individually Markov. Due to the anti-monotonicity of the transition rules, minimal and maximal process specify the bounding process

$$Y_{-s}^{-T} \;=\; \text{image}(F_{-s}^{-T}) \;=\; \bigcup\Big\{x \in E:\; Y_{-s}^{\min(-T)} \;\preceq\; x \;\preceq\; Y_{-s}^{\max(-T)}\Big\}.$$

Thus they are an efficient representation of the bounding process and can be used to monitor when coalescence has been achieved. The concept of a

cross-over was first developed in a more general context by Kendall [73], see also Section 5.4.3, and further examined by Häggström and Nelander [52].

Notice that an analogous construction can be used for continuous-time Markov chains; for an application see Example 5.9 or the algorithm developed in Chapter 6.

So far we have presented CFTP algorithms which seem only practical for infinite state spaces if we can

1. equip the state space with a partial order for which it has (not necessarily unique) maximal and minimal elements and

2. apply monotone or anti-monotone CFTP.

The next Section 5.4.3 will present a CFTP variant which by conditioning on the underlying process produces (random) maximal and minimal elements which enable us to monitor the bounding process. Section 5.4.4 then discusses algorithms for Markov chains which are not necessarily monotone but where the bounding process eventually collapses into a discrete set which can be efficiently represented.

5.4.3 Dominated CFTP

This variant of CFTP was first developed by Kendall [73], see also [71, 72]. In *dominated* CFTP the underlying process not only functions as a source of

randomness for the target Markov chain but also defines which states can be viewed as maximal or minimal. This produces maximal and minimal states which are stochastically varying over time. (Note that these maximal and minimal states do not necessarily have to lie within the state space, see for example the algorithm presented in Chapter 6.) Suppose that $h_0(U_{-t})$ defines the minimal state and $h_1(U_{-t})$ the maximal state at time $-t \leq 0$, where h_0 and h_1 are deterministic functions on the state space $\mathcal{U}$. We can construct a dominated CFTP algorithm, if we can find a functional H of the underlying process U such that

$$Y_{-s}^{-T} = \bigcup \Big\{ H(\{U_{-t}, s \leq t \leq T\}, x) \ : \ h_0(U_{-T}) \leq x \leq h_1(U_{-T}) \Big\}, \qquad (42)$$

is not only bounding process satisfying requirements (33) – (36) but a set which can be efficiently represented. Notice that in contrast to the definition in (32) on page 100, the functional H is dependent on an initial state x, which lies between the (random) minimal and maximal state. All existing dominated CFTP algorithms [71, 72, 73, 74, 75, 95, 98] rely on such a functional H to ensure that the bounding process can be efficiently represented.

The algorithms in [71, 72, 73, 74, 75, 95] use dominated CFTP together with monotone or anti-monotone CFTP. The birth-death process example 5.5 is an illustration of how dominated CFTP can be used together with monotone CFTP. Here the minimal state is 0 and thus non-random but the maximal state at any time $-t$ is given by Z_{-t}. The first article [73]

introducing dominated CFTP showed how to produce perfect samples of both the attractive and the repulsive area-interaction process in a very similar fashion to Example 5.5. Here is a short description of the algorithm.

Example 5.9 (Area-interaction process) *Each birth time $-t$ of a constant rate spatial birth-and-death process Z is marked with an independent uniform $[0,1]$ random variable M_{-t}. The marked spatial birth-and-death process functions as an underlying process. A process X which converges to the distribution of an area-interaction process is produced by thinning the births of Z. We start X at time $-T$ from the point pattern Z_{-T} and evolve it in accordance with Z. As Z is started in equilibrium, it is time-homogeneous. If at time $-t$ a point x is born in Z the birth of the same point is allowed to take place in X if and only if*

$$M_{-t} \quad \leq \quad \lambda(X_{-t-},\, x)$$

where $\lambda(\cdot,\cdot)$ is the Papangelou conditional intensity of the target area-interaction process as given in Example 2.12 on page 23. Deaths of Z are always allowed to take place in X if the dying point exists in X. Let the functional G describe the above procedure. The subset relation is used as a partial order on the state space of finite point configurations. If the area-interaction process is attractive, then the greater the point pattern with respect to the subset relation the more likely a birth is accepted and so G is monotone. We can define a

bounding bounding process satisfying the necessary requirements as

$$Y_{-s}^{-T} = \bigcup_{\emptyset \subseteq x \subseteq Z_{-T}} G\big(\{U_{-t}, s \le t \le T\};\, x\big).$$

A minimal and a maximal process are given by

$$Y_{-s}^{\min(-T)} = G\big(\{U_{-t}, s \le t \le T\};\, \emptyset\big)$$

and

$$Y_{-s}^{\max(-T)} = G\big(\{U_{-t}, s \le t \le T\};\, Z_{-T}\big).$$

Now consider a repulsive area-interaction. Here the acceptance of births is anti-monotone, that is the greater the configuration of X_{-t-} the less likely a birth is accepted at time $-t$. Thus the definition of minimal and maximal process needs to be based on a cross-over as described in Section 5.4.2. As before we start the minimal process at time $-T$ in the empty point configuration and the maximal process in Z_{-T}. Similar to X we evolve both processes in accordance with Z. However, the decision if a birth is allowed to take place in the minimal process depends on the Papangelou conditional intensity of the maximal process and vice versa for the maximal process. In this way we produce the cross-over needed for anti-monotone CFTP.

A further example of dominated and anti-monotone CFTP can be found in Chapter 6.

5.4.4 Set-orientated CFTP

Set-orientated CFTP is useful for MCMC algorithms which update a large collection of states to a single new state, thus producing a bounding process which eventually becomes a (small) discrete set. This CFTP variant does not require any monotonicity properties of the transition kernel and thus might be applicable when monotone or anti-monotone CFTP is not posssible.

Murdoch and Green [97] were the first to consider set-orientated CFTP, which they developed for continuous distributions in Bayesian inference problems. We illustrate this CFTP variant using the multigamma coupler from [97]. For simplicity assume that the state space $E = \mathbb{R}$.

Example 5.10 (Multi-gamma Coupler) *Suppose the transition kernel density $p(\cdot|x)$ of the target Markov chain X is such that $p(y|x) \geq r(y)$ for all $x, y \in E$, where $r(\cdot)$ is a non-negative function with $\int r(y)dy = \rho > 0$. Set*

$$R(y) = \rho^{-1} \int_{-\infty}^{y} r(v)dv \quad \text{and} \quad Q(y|x) = (1-\rho)^{-1} \int_{-\infty}^{y} [p(v|x) - r(v)]dv.$$

The following transition rule draws from $p(\cdot|x)$:

$$f(x, U_{-n}^{(1)}, U_{-n}^{(2)}) \quad = \quad \begin{cases} R_1(U_{-n}^{(2)}) & \text{if } U_{-n}^{(1)} < \rho \\ Q^{-1}(U_{-n}^{(2)}|x) & \text{otherwise,} \end{cases}$$

where $\{(U_{-n}^{(1)}, U_{-n}^{(2)}), n \in \mathbb{N}\}$ is a sequence of independent pairs of uniform $[0,1]$ random variables. As usual, let $f_{-n}(\cdot) = f(\cdot, U_{-n}^{(1)}, U_{-n}^{(2)})$ and

$$F_{-s}^{-T} \quad = \quad f_{-(s+1)} \circ \cdots \circ f_{-T}.$$

A bounding process satisfying the necessary requirements is given by

$$Y_{-s}^{-T} \quad = \quad \text{image}\left(F_{-s}^{-T}\right).$$

Keeping track of Y_{-s}^{-T} for a continuous state space E is often infeasible, but in the case considered here it can be simplified as follows. Suppose $T \in \mathbf{N}$ is large enough so that there is at least one natural number $m \leq T$ with $U_{-m}^{(1)} < \rho$. This ensures that a regeneration occurs in the time interval $[-T, 0]$. Then $Y_{-m+1}^{-T} = R^{-1}(U_{-m}^{(2)})$ and so the regeneration is sufficient (but not neccesary) for the bounding process to become a singleton. Let

$$k \quad = \quad \max\{m \leq T : U_{-m}^{(1)} < \rho\}$$

and set

$$\check{Y}_{-s}^{-T} = \begin{cases} E & \text{if } -s \leq -k \\ R^{-1}(U_{-k}^{(2)}) & \text{if } -s = -k+1 \\ F_{-s}^{-k+1}(\check{Y}_{-k+1}^{-T}) & \text{for } -s > -k+1. \end{cases}$$

This also defines a bounding process but is much easier to implement than the original bounding process. Before and at time $-k$ the process $\check{Y}_{-s}^{-T}$ is equal to the state space E, then at time $-k+1$ it becomes a singleton. As $U_{-m}^{(1)} < \rho$ occurs almost surely in finite time, the bounding process will collapse to a singleton in finite time. Note that it might take longer for $\check{Y}_{-s}^{-T}$ to coalesce than for the original bounding process, but the simplicity of the former outweighs this disadvantage. Furthermore, notice that we only need

to sample $(U_{-n}^{(1)}, U_{-n}^{(2)})$ *for* $n \leq T$, *where* $T = \min\{m \in \mathbb{N} : U_{-m}^{(1)} \leq \rho\}$. *Then* $\check{Y}_0^{-T}$ *is a singleton and so we only need one CFTP iteration.*

Set-orientated CFTP relies on the fact that the bounding process eventually becomes a discrete set which can be efficiently represented and monitored. As in the example of the multigamma coupler it is not necessary that the bounding process is the smallest possible bounding process. Much more important is that we can efficiently update it. If we do not choose the smallest possible bounding process then the runtime of the CFTP algorithm (in terms of number of transitions) will increase. However, choosing an adequate bounding process might substantially reduce computational complexity of each update and thus make each individual transition faster. Furthermore, it will simplify checking whether the bounding process has become a singleton.

In [97] the multigamma coupler presented in Example 5.10 is further refined to increase applicability. Moreover, a set-orientated CFTP algorithm is devoloped for a Metropolis-Hastings algorithm. This requires a proposal density which only takes a finite number of values. In [98] the concept is extended to a random walk Metropolis algorithm with a symmetric proposal density and combined with dominated CFTP.

The multigamma coupler considered in [97] is only defined for chains for which the whole state space is a small set and thus for transition kernels which are uniformly ergodic, see Proposition 4.6. Corcoran and Tweedie [20]

extended the multigamma coupler to a CFTP algorithm for Harris recurrent chains.

Finally, set-orientated CFTP for Markov random fields is explored in [53]. Here, a set-orientated approach allows efficient updates of the bounding process in cases where monotone or anti-monotone CFTP cannot be applied.

5.4.5 Occlusion CFTP

For some Markov chains the bounding process is produced by an occlusion procedure. In this section we will present examples where the occlusion procedure leads to a bounding process which can be represented by a single path. In these cases the CFTP algorithm is not (much more) complex than the analogous conventional MCMC algorithm. An example from stochastic geometry is the algorithm for the dead leaves model by Kendall in [75]. We have rephrased the example to show that it is an application of Theorem 5.2. For a description of the dead leaves model see Section 2.2.3. The dead leaves model is defined as the equilibrium distribution of the following Markov chain. Suppose we have a Poisson process of intensity θ of space-time points in $W \times [0, \infty) \subset \mathbf{R}^2 \times [0, \infty)$, where W is a bounded window. Each space-time point (x_i, t_i) (where $0 \leq t_1 < t_2 < \ldots$) is marked with a grain Ξ_i which is an independent copy of the primary grain Ξ. This marked Poisson process

parametrizes the space-time Boolean model $\Theta = \{\Theta_t, t \geq 0\}$, where

$$\Theta_t \quad = \quad \bigcup_{0 \leq t_i \leq t} \left(x_i \oplus \Xi_i \right).$$

We assume that Θ covers the window W in almost surely finite time. Consider now the process $\{\Psi_t : t \geq 0\}$ which describes the evolution of the tessellation produced by the space-time Boolean model. At time $t = 0$ we start with the empty set, that is $\Psi_0 = \emptyset$. Whenever a space-time point is born, we add to Ψ_t the boundary $\partial\Xi_i$ of the associated grain and take away the parts of boundaries of previous grains covered by the new grain. This means Ψ_t changes only at time points $\{t_i\}$, with

$$\Psi_{t_i} \quad = \quad \partial\Xi_i \cup \left(\Psi_{t_i -} \setminus \operatorname{int}(\Xi_i)\right), \tag{43}$$

where $\Psi_{t_i -}$ is the configuration of Ψ immediately prior to time t_i. (Figure 14 illustrates a single step of this construction.) The equilibrium distribution of $\{\Psi_t : t > 0\}$ is the dead leaves model.

The following algorithm describes how to derive a realisation of the tessellation process Ψ from a realization u of the marked space-time Poisson process on some time interval $[S, T]$.

```
Tessellation(u, S, T):
   Ψ ← ∅
   t ← S
   repeat
      (x, t, Ξ) ← first marked incident of u after t
      Ξ ← Ξ ⊕ x
```

```
    Ψ ← ∂Ξ ∪ (Ψ \ int(Ξ))
    until (x, t, Ξ) last marked incident of u before time T
  return Ψ
```

Suppose that `PoissonPointProcess`$(\theta, W \times [0,T], \Xi)$ delivers a Poisson point process of intensity θ over $W \times [0,T]$. Suppose further, that each incident of the produced realisation of the Poisson process is marked with an independent random set distributed as Ξ. Then the following algorithm produces an (approximate) sample of the dead leaves model:

```
ForwardDeadLeaves(T, θ, W, Ξ):
  z ← PoissonPointProcess(θ, W × [0,T], Ξ)
  Ψ ← Tesselation(z, 0, T)
  return Ψ
```

But how do we choose T such that the output of `ForwardDeadLeaves` is a sample of the dead leaves model? If we stop at a fixed time T then there is a positive probability that the window W is not yet covered. Thus the sample cannot be a sample from the dead leaves model, which would produce complete coverage with probability one. If we instead stop at the first time T that complete coverage is achieved, then the sample might be biased towards configurations which cover W efficiently, for further comments see Section 5.5.2. Thus the above algorithm can be implemented only in an approximate form. We will now show how CFTP can be applied to this setting.

Example 5.11 (Dead leaves model) *Let Z be a Poisson process of space-*

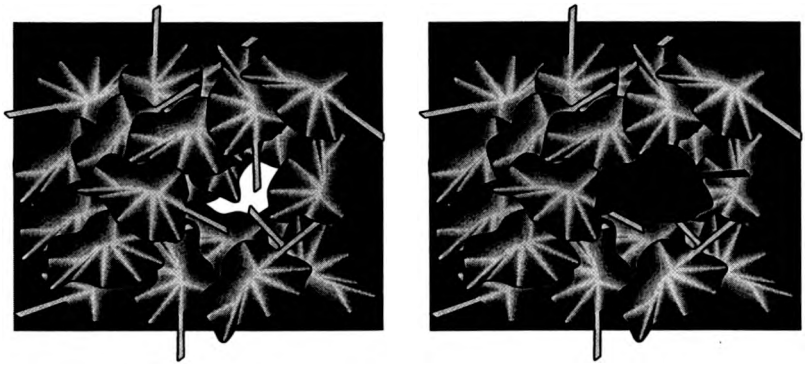

Figure 14: Simulated dead leaves model. Careful inspection shows that the image on the left still has an uncovered portion. The image on the right is obtained using one further grain, which happens to cover the previously uncovered portion. Continuing this simulation will further alter the image, as new leaves are superimposed on top of the previous pattern.

---

time points of intensity θ in $W \times (-\infty, 0]$. We now mark each space-time point $(x_i, -t_i)$ of Z (where $0 \geq -t_1 > -t_2 > \ldots$) with an independent (random) grain Ξ. The resulting marked space-time Poisson process U, which can be easily extended backwards in time, will function as our underlying process in the CFTP algorithm. Furthermore, we can produce a process Ψ^{-T}, which behaves like Ψ started at time $-T$ in the empty set, as an adapted functional of $\{U_{-t}, -t \geq -T\}$; we just need to apply the procedure `Tesselation` *to the realisation of $\{U_{-t}, -t \geq -T\}$.*

We now define the bounding process. Let

$$Y_{-s}^{-T} \quad = \quad \left\{\Psi_{-s}^{-u} : -u \leq -T\right\},$$

be a set of tessellations. (Note that this is actually a countable set as Ψ_{-s}^{-u} only changes for a countable set of u.) We will first show that this bounding process satisfies all requirements and then discuss how it can be efficiently represented. Firstly, the funneling property (33) is satisfied because the union of tessellations forming Y_{-s}^{-T-S} for $-S \leq 0$ will have less elements then the union forming Y_{-s}^{-T}. Secondly, if the bounding process becomes a singleton, which is to say its component tessellations are identical, it must remain a singleton and thus it satisfies the first coalescence property (34). Now, observe that Y_0^{-T} is a singleton if and only if the random set

$$\Theta_0^{-T} \quad = \quad \bigcup_{(x_i,-t_i,\Xi_i)\in U_{[-T,0]}} \left(x_i \oplus \Xi_i \right)$$

covers the whole window W. As we assumed that this occurs in almost surely finite time, the bounding process becomes a singleton in almost surely finite time and so satisfies the second coalescence property (35). Finally, the sandwiching property (36) is satisfied because

$$\Psi_{-s}^{-T} \quad \in \quad Y_{-s}^{-T},$$

where $\Psi_{-T}^{-T} = \emptyset$. As mentioned above, Y_0^{-T} becomes a singleton if Θ_0^{-T} covers the whole window. By the definition of the bounding process it then follows that the singleton Y_0^{-T} is equal to Ψ_0^{-T}. A naive CFTP approach now leads to the following procedure. Suppose the algorithm `Extend`$(u, -T)$ *produces a realisation of U on $[-T, 0]$ by extending the realisation u backwards in time.*

```
DeadLeavesCFTP(θ, W, Ξ):
   T ← 0
   u ← ∅
   Repeat
      T ← T + 1
      u ← Extend(u, −T)
      Θ ← ∪_{(x,t,Ξ)∈u}(x ⊕ Ξ)
      until Θ covers W
   Ψ ← Tesselation(u, −T, 0)
   return Ψ
```

However, the above procedure can be much simplified by observing the following. Firstly, we can determine the coalescence time T_C by starting the process U at time 0, simulating U backwards in time and simultaneously simulating the random set

$$\tilde{\Theta}_{-t} \quad = \quad \bigcup_{(x_i,-t_i,\Xi_i)\in U_{[-t,0]}} \left(x_i \oplus \Xi_i \right)$$

until the set $\tilde{\Theta}_{-t}$ covers the whole window. Furthermore, we can produce the exact sample $\Psi_0^{-T_C}$ by simulating backwards instead of forwards in time. Suppose we have a realisation of the underlying marked Poisson process U on $[-T, 0]$ with ordered *negative* time-points $0 \geq -t_1 > -t_2 > \ldots > -t_n \geq -T$. Now, consider the following backwards tessellation process $\{\tilde{\Psi}_{-t}, -t \leq 0\}$. Similar to before, we set $\tilde{\Psi}_0 = \emptyset$ and changes in $\tilde{\Psi}_{-t}$ occur only at time points $\{-t_i\}$. But now each time a grain is laid down, we record the part of

its boundary that is not covered by previous grains. Thus

$$\bar{\Psi}_{-t_i} \quad = \quad \bar{\Psi}_{-t_i+} \cup \left((\partial \Xi_i) \backslash \bigcup_{j<i} \mathrm{int}(\Xi_j) \right), \tag{44}$$

where $\bar{\Psi}_{-t_i+}$ is the configuration of $\bar{\Psi}$ immediately subsequent to time $-t_i$. Figure 15 illustrates the construction. Like $\bar{\Theta}$ the backwards tessellation process $\bar{\Psi}$ can be simulated simultaneously to U. The backwards process $\bar{\Psi}$ differs from the forward process Ψ merely by a point of view. Informally, the conventional forwards simulation views the ground from above as it becomes covered with dead leaves. The backwards simulation views the same process from beneath the ground, as if one had adopted the point of view of a small animal looking upwards from a hole in the ground! This difference is illustrated in an animated simulation which can be found on the website

`http://www.warwick.ac.uk/statsdept/Staff/WSK/dead.html`.

We can move from a backwards realisation to a forwards realisation simply by reversing the order in which the leaves are put down. For example the realisation in Figure 15 was produced by using the same leaves as in Figure 14, but placed in reverse order. More formally, $\bar{\Psi}$ is the time reversal of Ψ. Thus if we use the same realisation of U to produce Ψ_0^{-T} and $\bar{\Psi}_{-T}$ then the two tessellations coincide. It follows that we can omit any forwards simulation in the CFTP algorithm and merely use backwards simulation to produce an exact sample of the dead leaves model.

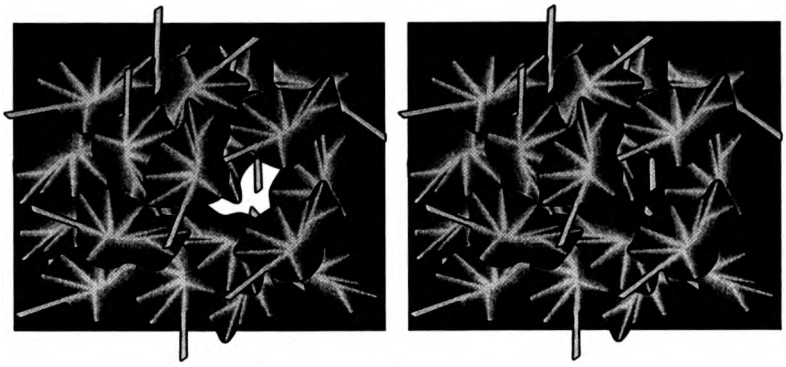

Figure 15: Perfect simulation of dead leaves model. The image on the left still has an uncovered portion. The image on the right is obtained using one further grain, which happens to cover the previously uncovered portion. In contrast to the conventional construction, however, the new leaf is added at the bottom, corresponding to simulation backwards in time and thus to perfect simulation. Clearly the backwards simulation can now be continued indefinitely without altering the resulting image, which therefore must be a sample from the equilibrium distribution.

A CFTP algorithm solely based on backwards simulation translates into pseudo-code notation as follows.

```
BackwardDeadLeaves(θ, W,Grain):
   Π ← PoissonPointProcess(θ, W × (−∞, 0))
   Ψ ← ∅
   Φ ← ∅
   t ← 0
   repeat
      (x, t, Ξ) ← first marked incident of Π before t
      Ξ ← Ξ ⊕ x
      Ψ ← Ψ ∪ (∂Ξ\Φ)
      Φ ← Φ ∪ Ξ
      until W ⊆ Φ
   return Ψ
```

Clearly, we could continue the construction backwards in time indefinitely and this would still produce the same output. We therefore have a *virtual simulation from time* $-\infty$, but it actually terminates after a finite random time. Thus we have a perfect sample from the equilibrium distribution, that is from the dead leaves model.

Propp and Wilson [113] also developed an example where the bounding process is produced by an occlusion procedure. This CFTP algorithm generates a random rooted spanning tree of a strongly-connected, directed, finite graph G whose arcs are marked with positive weights.

Example 5.12 (Random rooted spanning trees) *A random walk on the set of rooted spanning trees is coupled with a random walk M on the graph.*

Each excursion of M from the root back to the root can be translated into a transition from a rooted spanning tree to another rooted spanning tree.

The algorithm starts with an array which is indexed by the vertices of G. Initially all entries of the array are set to `nil`. *The random walk on G is started in the root and observed until it returns to the root. For each vertex which is visited during this excursion, we enter the next visited vertex into the array. We repeatedly observe the random walk on G and enter or overwrite entries of the array. Besides recording the excursions of M, the array also functions as a bounding process on the set of rooted spanning trees of G. If the random walk on the graph has visited all vertices of G, the array will have an entry at every index except at the root. It follows that the bounding process is a singleton, thus we can use this procedure for a CFTP algorithm.*

Similarly to the dead leaves example, the algorithm simplifies if we can simulate the time reversal $\bar{M}$ of M. Then simultaneously simulating $\bar{M}$ and the spanning tree random walk backwards in time until $\bar{M}$ has visited all vertices produces a random rooted spanning tree. This latter procedure using the time-reversal of M had been developed for special cases [2, 13, 68] even before CFTP was known.

5.5 Further Results and Comments

The first part of this section summarizes some theoretical results on the runtime of CFTP. Thereafter we discuss some aspects which are essential for a correct implementation of CFTP.

5.5.1 Runtime

Consider a discrete-time, ergodic Markov chain on a partially ordered finite state space $(E, \preceq)$ with unique minimal element $\hat{0}$ and unique maximal element $\hat{1}$. Propp and Wilson [111] relate the mixing time of the chain to the runtime of monotone CFTP. Using the same notation as in Section 5.4.1, let

$$T_{MC} \quad = \quad \min\{T \geq 0 : F_0^{-T}(\hat{0}) = F_0^{-T}(\hat{1})\}$$

be the *backward coalescence time* for monotone CFTP. Define

$$d(k) \quad = \quad \max_{\mu_1, \mu_2} ||\mu_1^k - \mu_2^k||,$$

where μ^k denotes the distribution of the Markov chain at time k if started in distribution μ at time 0. Suppose l is the length of the largest totally ordered subset in E. Then [111, Theorem 5]

$$l^{-1}\, \mathbf{P}(T_{MC} > k) \quad \leq \quad d(k) \quad \leq \quad \mathbf{P}(T_{MC} > k).$$

Furthermore, [111, Lemma 7]

$$k\, \mathbf{P}(T_{MC} > k) \quad \leq \quad \mathbf{E}(T_{MC}) \quad \leq \quad \frac{k}{\mathbf{P}(T_{MC} \leq k)}.$$

Now define the *mixing time threshold* $T_{\text{mix}} = \min\{k : \ d(k) \leq e^{-1}\}$, then

$$\mathbf{E}(T_{MC}) \quad \leq \quad 2\, T_{\text{mix}}(1 + \ln l).$$

Thus "if a Markov chain is rapidly mixing then it is rapidly coupling" [111]. If a rigourous (and practical) bound on the mixing time threshold is available, then conventional Markov chain Monte Carlo would usually run the chain for a multiple of T_{mix} to ensure that the chain is sufficiently close to equilibrium. However, using CFTP ensures that the chain is exactly in equilibrium and has an expected runtime which is also a multiple of T_{mix}.

Monotone CFTP is based on finding the time T_{MC} by going iteratively further in the past. It would be very inefficient, if we would check for each T whether $T = T_{MC}$. More sensible seems to be the approach of starting with some $T_0 > 0$ and then checking whether $T_0 \geq T_{MC}$. For this we need to check whether the minimal and the maximal process when started from time $-T_0$ agree at time 0. If $T_0 < T_{MC}$ we try time $T_1 = rT_0$ for some constant $r > 1$ and after this $T_2 = rT_1$, and so on. Let k be the smallest integer such that $r^k T_0 \geq T_{MC}$, then the number of transition steps required to find k is $2\, T_0\, (1 + r + \ldots + r^k)$, where the factor 2 takes account of the fact that we produce a path for both the minimal and for the maximal process. On the other hand, even if we could guess T_{MC}, in order to verify T_{MC} we would

need 2 T_{MC} transitions. Now, notice that

$$2\,T_0\,(1+r+\ldots+r^k) \;=\; 2T_0\frac{r^{k+1}-1}{r-1} \;<\; 2T_0 r^{k-1}\frac{r^2}{r-1} \;\leq\; 2T_{MC}\frac{r^2}{r-1}.$$

Thus the ratio between the number of transitions when using the former strategy and the number of transitions for a right guess is bounded above by $r^2/(r-1)$ which is minimized for $r=2$, where it takes value 4. Thus Propp and Wilson [111] recommend to use $T_n = -2^n, n \in \mathbb{N}$, a strategy which has been widely adopted. The above argument is taken from [111].

Foss and Tweedie [41] put CFTP into the context of stochastic recursive sequences as developed by Borovkov and Foss [11, 12]. Stochastic recursive sequences are based on the construction of a probability space $(\Omega, \mathcal{F}, \mathbb{P})$, a sequence $\{\xi_n, n \in \mathbb{Z}\}$ of independent and identically distributed random variables on a state space $\mathcal{U}$ and a measureable function $f : E \times \mathcal{U} \to E$ such that the chain X can be produced by setting

$$X_0 \;=\; x_0, \qquad X_{n+1} \;=\; f(X_n, \xi_n), \qquad n \geq 0. \tag{45}$$

Clearly the function f is a transition rule and the sequence $\{\xi_n, n \in \mathbb{Z}\}$ can be used as an underlying process in a CFTP algorithm. Let θ^m denote the m-shift transformation, that is the map $\theta^m : \Omega \to \Omega$ such that for any sequence $\{\omega_n\}_{n\in\mathbb{N}}$ in Ω we have $\theta^m(\{\omega_n\}_{n\in\mathbb{N}}) = \{\omega_{n+m}\}_{n\in\mathbb{N}}$. For any random variable Y we then have $\theta^m(Y(\omega)) = Y(\theta^m(\omega))$. Let

$$g_n(x, \xi_0, \ldots, \xi_{n-1}) \quad = \quad f(\cdots f(f(x, \xi_0), \xi_1)\cdots), \xi_{n-1}).$$

Then for the Markov chain X in (45) we have

$$\theta^m X_n \quad = \quad g(\theta^m X_0, \xi_m, \ldots, \xi_{m+n-1}).$$

A *minimal backward coupling time* is defined as

$$T_{BC} = \min\{m \geq 0 : \theta^{-n_1} X_{n_1} = \theta^{-n_2} X_{n_2},\ \forall n_1, n_2 \geq m\}.$$

A *vertical backward coupling time* is defined as

$$T_{VC} \quad = \quad \min\{n \geq 0 : \theta^{-n} X_n^{(x)} = \theta^{-n} X_n^{(y)},\ \forall x, y \in E\},$$

where $X_n^{(x)}$ is the chain X at time n if started at time 0 in the non-random state x. The authors show the following Proposition:

Proposition 5.13 *There is a successful (that is, almost surely finite) vertical backward coupling time if and only if the chain X is uniformly ergodic.*

Thus the original CFTP algorithm developed in [111] is only applicable to uniformly ergodic chains. However, dominated CFTP methods, like the algorithms in [71, 72, 73, 74, 75], are possible for chains which are not uniformly ergodic.

We give a short outline of the proof of Proposition 5.13 as given in [41]. First observe that T_{VC} is submultiplicative, that is

$$\mathbf{P}(T_{VC} > m+n) \quad \leq \quad \mathbf{P}(T_{VC} > m)\ \mathbf{P}(T_{VC} > n)$$

and thus $\mathbb{P}(T_{VC} > n) \leq c\lambda^n$ for some $c \geq 0$ and $0 < \lambda < 1$. The vertical backward coupling time is governed by the same law as the forward coupling time

$$T_{FC} \quad = \quad \min\{n \geq 0 : X_n^{(x)} = X_n^{(y)} \ \forall x, y \in E\}.$$

With the coupling inequality (24) it follows that

$$||\mathcal{L}(X_n) - \pi|| \quad \leq \quad 2\mathbb{P}(T_{FC} > n) \quad \leq \quad 2c\lambda^n.$$

On the other hand, if the chain is uniformly ergodic then there is an $m \in \mathbb{N}$ such that the m-step transition kernel is uniformly minorized with minorization constant ρ. Thus we can define a CFTP multigamma coupler, as in Example 5.10, for $\{X_{nm}, n \geq 0\}$ and deduce that there is a succesful vertical coupling time with

$$T_{VC} \quad \leq \quad \min\{nm \geq 0 : \ U_{-nm}^{(1)} \leq \rho\} \quad < \quad \infty \quad \text{almost surely,}$$

where $\{U_{-nm}^{(1)}, -n \leq 0\}$ is a sequence of independent uniform $[0,1]$ random variables.

Suppose $\check{X}$ is a stationary version of X started in $\check{X}_0 = x_0$, where x_0 is a draw from π, and produced by setting $\check{X}_{n+1} = f(X_n, U_n)$, where $f(\cdot, U_n)$ is a transition rule for the transition kernel of P. Suppose X' is started in an arbitrary initial state and coupled to $\check{X}$ through the transition rule f. Now, consider the following forward coupling time

$$\check{\tau} \quad = \quad \min\{n \geq 0 : \check{X}_n = X'_n\}.$$

Foss and Tweedie [41] show that a minimal backward coupling time T_{BC} loses at most one polynomial moment from $\check{\tau}$ and has the same order of geometric convergence as $\check{\tau}$.

The next section points out some aspects which ensure a correct implementation of CFTP.

5.5.2 Forward Coupling

We would save a lot of computational overhead if we based the coupling procedure on a forward rather than a backward simulation. If we simulate coupled sample paths from all initial states forwards in time until we achieve complete coalescence then the current state of the chain is independent of the initial state and thus the chain is in equilibrium. However, the state of the chain at the time of coalescence is not necessarily independent of the time of coalescence and thus might not be a sample of the equilibrium distribution. Here is an example how forward coupling can lead to a biased sample.

Example 5.14 (Random walk on 3 states) *Consider the random walk X on the state space $\{0,1,2\}$ with transition matrix*

$$P = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} & 0 \\ 0 & 0 & 1 \\ \frac{1}{2} & 0 & \frac{1}{2} \end{pmatrix}.$$

The stationary distribution of X *is given by* $\pi_0 = 2/5$, $\pi_1 = 1/5$ *and* $\pi_2 = 2/5$. *We can simulate* X *using the following transition rule:*

for $x = 0, 2$

$$f(x, U) = \begin{cases} 0 & \text{if } U \leq 1/2 \\ \min\{x+1, 2\} & \text{if } U > 1/2 \end{cases}$$

for $x = 1$

$$f(1, U) = 2,$$

where U *is a uniform random variable on* $[0, 1]$.

If we simulate coupled paths of the chain started in each of the states $\{0, 1, 2\}$ *and using the above transition rule, coalescence cannot occur after the first transition step. After the first transition step the possible states are* $\{0, 2\}$ *or* $\{1, 2\}$. *Moreover until coalescence occurs the set of possible states alternates between* $\{0, 2\}$ *and* $\{1, 2\}$. *In either case the probability of coalescence in the next step is 1/2. See Figure 16 for an illustration. At the time of coalescence the chain will be either in state* 0 *or* 2 *each with probability 1/2. Thus if we sample* X *at the time of coalescence we will not sample the equilibrium distribution.*

A further example illustrating why simulation backwards in time is essential for CFTP is given by the dead leaves model.

Example 5.15 (Dead Leaves) *Whereas backwards simulation is unbiased if we stop the algorithm at the first time of complete coverage, forward sim-*

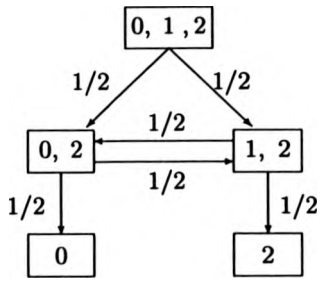

Figure 16: Illustration of the forward transitions until coalescence for the random walk in Example 5.14.

---

ulation will lead to a subtle bias if the simulation is run only till the first time the window is covered by grains. To illustrate this consider the following simplified example. Assume we obtain random grains $\Xi_1, \ldots, \Xi_n$ *at times* $-t_1 > \ldots > -t_n$ *in the backwards simulation such that the last grain* Ξ_n *covers the whole window* W*. To allow the patterns of backward and forward simulation to agree we assume the grains of the forward simulation to be* $\Xi_n, \ldots, \Xi_1$ *at times* $0 = t_n - t_n < t_n - t_{n-1} < \ldots < t_n - t_1$*. As* Ξ_n *covers all of* W*, complete coverage occurs in the forward simulation before time* $t_n - t_1$*, in fact already at time 0. This provides the intuition explaining the subtle bias mentioned above: in the case that a single grain can cover the whole window there is an increased possibility of single-grain coverage.*

5.5.3 Backwards Extension

It is important that whenever we go a step backwards in time we do not produce a completely new realisation of the underlying process, but extend

the previous realisation backwards in time. Only then the CFTP output is guaranteed to be a sample from the equilibrium distribution.

Example 5.16 (Random walk on 2 states) *Consider the Markov chain Y on $\{0,1\}$ with transition matrix*

$$P = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} \\ 1 & 0 \end{pmatrix}.$$

This Markov chain, which was also considered in [37] and [71], has equilibrium distribution $(\pi_0, \pi_1) = (2/3, 1/3)$. The following function is a transition rule for Y:

$$f(0,U) = \begin{cases} 0 & \text{if } U \leq 1/2 \\ 1 & \text{if } U > 1/2 \end{cases}$$
$$f(1,U) = 0,$$

where U is a uniform variable on the unit interval. Now suppose we use Voter CFTP to produce a sample from π. However, in each iteration we do not re-use the already sampled transition rules of the previous iteration but sample independent ones. We assume that in each iteration we go one step further back into the past such that the run of the k^{th} iteration has k transitions. Let Z describe the output of this incorrect CFTP procedure, then

$$\begin{aligned} \mathbf{P}\big(Z=0\big) &= \sum_{k=1}^{\infty} \mathbf{P}\big(\textit{ first coalescence in kth iteration and } Z=0\big) \\ &= \sum_{k=1}^{\infty} (1/2)^{k(k-1)/2} \sum_{j=1}^{k} (1/2)^j P^{k-j}(0,0) \end{aligned}$$

$$
\begin{aligned}
&= \sum_{k=1}^{\infty} (1/2)^{k(k-1)/2} \sum_{j=1}^{k} (1/2)^j \, 2/3 \, (1 - (-1/2)^{k-j+1}) \\
&= 2/3 \sum_{k=1}^{\infty} (1/2)^{k(k-1)/2} \, (1 - (1/2)^k + (1/2)^{k+1} I_{(\ k \text{ odd }\ )}) \\
&> 2/3
\end{aligned}
$$

Clearly, we do not sample the equilibrium distribution.

5.5.4 User Impatience Bias

Note that the running time of CFTP in terms of Markov transitions and the state sampled by the algorithm are not independent. Hence an impatient user who aborts long runs may introduce a subtle bias. As an example consider again random walk on three states in Example 5.14.

Example 5.17 (Random walk on 3 states) *Suppose a users runs CFTP to produce a sample of the equilibrium distribution of the random walk defined in Example 5.14. The user is impatient and terminates a run of CFTP whenever I iterations have been completed without producing output. Conditional on obtaining output the user will not sample output $Z \in \{0, 1, 2\}$ according to the equilibrium distribution as the following calculations show. We adopt Propp and Wilson's recommendation [111] and set the length of the run in the i^{th} iteration of CFTP equal to 2^{i-1}. Let $N = 2^{I-1}$ be the number of transitions in the last iteration of a CFTP run. Conditional on coalescence within the first I iterations the user will sample output Z with the following*

probabilities:

$$\mathbf{P}\Big(Z = r \,\Big|\, I \Big) = \frac{\mathbf{P}\Big(\textit{coalescence within } I \textit{ iterations and } Z = r \Big)}{\mathbf{P}\Big(\textit{coalescence within } I \textit{ iterations} \Big)} = \frac{\sum_{k=2}^{N} (\frac{1}{2})^{k-1} \Big[\frac{1}{2} P^{(N-k)}(0,r) + \frac{1}{2} P^{(N-k)}(2,r) \Big]}{1 - (\frac{1}{2})^{N-1}}.$$

Now observe that if

$$a = (1,0,-1), \qquad b = (0,1,-1), \qquad c = (1,1/2,1),$$

then

$$a\,P = 1/2\,b, \quad b\,P = -1/2\,a \quad \textit{and} \quad c\,P = c.$$

It follows that for $k \in \mathbf{N}$

$$(1,0,1)\,P^{2k} = 1/5\Big[(-1)^k (1/2)^{2k} a + (-1)^{k+1} (1/2)^{2k-1} b + 4c\Big]$$

$$(1,0,1)\,P^{2k+1} = 1/5\Big[(-1)^{k+1} (1/2)^{2(k-1)} a + (-1)^{k+1} (1/2)^{2k-1} b + 4c\Big],$$

Using these expressions we obtain for $I = 2$*:*

$$\mathbf{P}\Big(Z = 0 \,\Big|\, I = 2\Big) = 1/2 \qquad \mathbf{P}\Big(Z = 2 \,\Big|\, I = 2\Big) = 1/2,$$

and for $I \geq 3$*:*

$$\mathbf{P}\Big(Z = 0 \,\Big|\, I\Big) = \frac{2}{5}\Big[\frac{1 - 2^N}{1 - 2^{-N+1}}\Big]$$

$$\mathbf{P}\Big(Z = 1 \,\Big|\, I\Big) = \frac{1}{5}\Big[\frac{1 - 2^N}{1 - 2^{-N+1}}\Big]$$

$$\mathbf{P}\Big(Z = 2 \,\Big|\, I\Big) = \frac{2}{5}\Big[\frac{1 - 7 \times 2^{N-1}}{1 - 2^{-N+1}}\Big].$$

Although the bias reduces with increasing maximum number of iterations I, the sample will always be biased.

Not all Markov chains suffer bias from user impatience. For instance for the Markov chain in Example 5.16 protection against this kind of bias is automatic as long as the number of transitions in the i^{th} CFTP iteration is 2^{i-1}.

Example 5.18 (Random walk on 2 states) *Consider the Markov chain defined in Example 5.16 and assume that the user sets the run in the k^{th} iteration of CFTP equal to 2^{k-1}. If the user allows for more than one iteration of CFTP then we have no user impatience bias as the following calculations show. Let I be the maximum number of iterations the user wants to use and $N = 2^{I-1}$. Then conditional on achieving coalescence within I iterations the user will sample output $Z = 0$ with the following probability:*

$$\begin{aligned}
\mathbb{P}\big(Z=0 \,\big|\, I\big) &= \frac{\mathbb{P}(\text{ coalescence within } I \text{ iterations and } Z=0)}{\mathbb{P}(\text{ coalescence within } I \text{ iterations })} \\
&= \frac{\sum_{k=1}^{N} P^{(N-k)}(0,0) \times (\frac{1}{2})^k}{1-(\frac{1}{2})^N} \\
&= \frac{\sum_{k=1}^{N} \left(\frac{2}{3}(1-(-\frac{1}{2})^{N-k+1})\right) \times (\frac{1}{2})^k}{1-(\frac{1}{2})^N} = \frac{2}{3}.
\end{aligned}$$

For more examples illustrating the bias caused by user impatience and bounds on the magnitude of this bias see Sections 5.2 and 6.1 in [37]. For further comments on how to avoid user impatience bias see [37, 111, 112].

6 Coupling from the Past for a Conditional Boolean Model

In this chapter we will develop a CFTP algorithm which samples exactly a Boolean model which is conditioned to cover a finite set of locations with grains. Besides being of practical value for applications in areas like mining engineering, the algorithm is of methodological interest as it extends the range of perfect simulation methods. The algorithm combines dominated CFTP with anti-monotone CFTP as defined in the previous Chapter 5. But in contrast to existing dominated CFTP algorithms for point processes, see Example 5.9 and [71, 72, 74], which derive a bounding process by thinning births of the underlying spatial birth-and-death process, we present a new technique which builds the bounding process by extending the life times of particles beyond death times. More importantly, we define the bounding process on a state space which is larger than the target model state space. Augmenting the state space leads to an efficient representation of the bounding process which is an essential requirement for any practical CFTP algorithm. Overall, the algorithm developed in this chapter shows how to exploit the very general framework given in Theorem 5.2 and produce feasible CFTP algorithms.

6.1 Introduction

Problems arising in applied geology and mining engineering motivate the modelling of populations of (random) objects using *Boolean random sets*: a point pattern of *germs* is generated according to a Poisson point process and at each germ an object or *grain* is located which is an independent realization of a random compact set; see section 2.2.3 or [25, 93, 133].

Often experimental data is available which provides further information on the location of grains. One may know that certain locations are covered or alternatively not covered by grains, one may know the number of grains covering a certain location or one may know that certain connectivity relations hold. Simulations should incorporate this data by imposing adequate conditions on the Boolean model. Lantuéjoul [79] studies the simulation of such *conditional* Boolean models. Extending an algorithm originally proposed by Matheron, Lantuéjoul produces a spatial birth-and-death process which converges to the target conditional Boolean model exponentially fast in distribution.

We restrict our attention to Boolean models on a bounded window W whose primary grain is almost surely contained in a disc of fixed radius. Without loss of generality we assume that this radius is equal to one. We condition the Boolean model to cover each point in a set $\mathcal{C} = \{z_1, \ldots, z_k\} \subset W$ with grains. For a discussion on coverage probabilities of Boolean models

see Hall [55].

In the first section of this chapter we determine an underlying process U and derive a Markov chain X which converges to the target conditional Boolean model. Then, in Section 6.3 we construct a bounding process Y and develop a CFTP algorithm. Section 6.4 presents some properties of the algorithm and Section 6.5 gives a constructive proof that the bounding process contains a realisation of X. We proceed in Section 6.6 to show that our CFTP algorithm is a correct implementation according to Theorem 5.2. The final section of this chapter discusses some extensions and further work.

6.2 The Underlying Process

We start by constructing an underlying process U from which we can derive a Markov chain which converges to the target conditional Boolean model. Following Lantuéjoul [79], we use a spatial birth-and-death process which converges to the unconditional Boolean model and prohibit transformations which do not respect the covering condition. As we will show in Theorem 6.3 the resultant conditional spatial birth-and-death process then converges to the conditional Boolean model.

Consider first an unconditional Boolean model Ψ with germ intensity measure μ and grain distribution H on the space of primary grains $\mathcal{G}$. We derive a Boolean model evolving in time as follows. Consider a space-time

Poisson process of mean measure $\mu(\cdot) \times \text{Leb}(\cdot)$ on $W \times \mathbb{R}$. We assume that μ is positive on W. Each point of this process is marked independently with a (random) grain drawn from H and an Exponential variable of unit mean which represents the life time of the grain. Thus we obtain a marked space-time Poisson process Q on $W \times \mathbb{R} \times \mathcal{G} \times (0, \infty)$ based on space-time points

$$(x, s, G, l) \quad \in \quad W \times \mathbb{R} \times \mathcal{G} \times (0, \infty).$$

Here s represents the birth time and l represents the life time of germ-grain pair (x, G).

We now derive a time-evolving marked point process $Z = \{Z_t, t \in \mathbb{R}\}$, where Z_t is the list of the germ-grain pairs of Q alive at time t:

$$Z_t \quad = \quad \{(x, G) \, : \, (x, s, G, l) \in Q, \; s \leq t \leq s + l\}.$$

Notice that Z evolves as a birth-death process of objects, each object being a germ-grain pair. Also note that $\{Z_t, t \in \mathbb{R}\}$ parametrizes a space-time Boolean model $\{\Xi_t, t \in \mathbb{R}\}$, where:

$$\Xi_t \quad = \quad \bigcup_{(x,G) \in Z_t} (x \oplus G) \, .$$

The random set Ξ_t, the induced set of Z_t, is distributed according to the unconditional Boolean model Ψ, if Z is started in equilibrium. Figure 17 illustrates the relation between Q and Ξ.

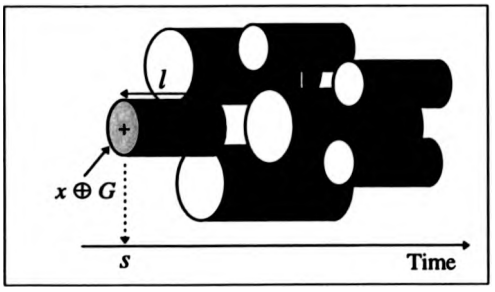

Figure 17: The marked space-time Poisson process Q parametrizes the cylinder model shown in this figure. The process Ξ describes the time-slices through the cylinder process.

We now alter Z allowing only transformations which respect the covering condition, that is we do not permit any transitions which uncover a point in C. Let $X = \{X_t : t \in \mathbb{R}\}$ denote this *conditional* spatial birth-and-death process. Note that we say a germ-grain pair (x, G) *covers a conditioning point* $z \in C$ if z is an element of the induced set $x \oplus G$. Clearly births never violate the covering condition, whereas deaths might uncover a conditioning point. Consequently, whenever a germ-grain pair is born in Z it is also born in X. If a germ-grain pair dies in Z the germ-grain pair becomes a candidate for death in X but in contrast to Z does not necessarily die. A death is only permitted if the removal of the germ-grain pair does not uncover a conditioning point, for an illustration see Figure 18. If a germ-grain pair survives a death time, we assign a new independent unit rate Exponential life time to it. At the end

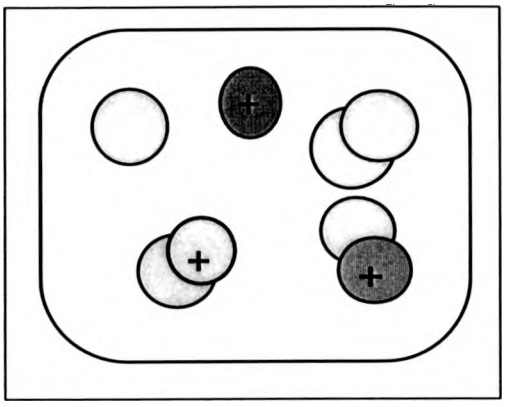

Figure 18: Perpetuation: the figure shows the conditioning points as crosses. The lighter shaded grains would be allowed to die in this configuration, the darker shaded grains would be perpetuated.

---

of this new life time we again check whether the germ-grain pair is allowed to die subject to the covering condition. We call a germ-grain pair which does not die at its earliest possible death time a *perpetuated* germ-grain pair and the new life time(s) assigned to it *perpetuation time(s)*.

The space-time marked Poisson process Z has a birth rate given by the measure $\mu(\cdot)$ and a unit death rate per grain, and so X has also birth rate $\mu(\cdot)$ but the death rate of X varies according to coverage. Like Z the process X parametrizes a space-time random set $\{\Theta_t, t \in \mathbb{R}\}$:

$$\Theta_t = \bigcup_{(x,G)\in X_t} (x \oplus G).$$

In the following we will often identify the process X with its induced

space-time random set $\{\Theta_t, t \in \mathbb{R}\}$ and the process Z with $\{\Xi_t, t \in \mathbb{R}\}$.

Let $(E, \mathcal{E})$ denote the state space of X. We now show that X converges to the target conditional Boolean model. To prove the result, we need the following definition of a petite set, see also Meyn and Tweedie [91].

Definition 6.1 (Petite Set) *Let Φ be a continuous-time Markov process with transition kernel $P^t(x, A) = \mathbb{P}(\Phi_t \in A | \Phi_0 = x)$. For some probability measure a on $\mathbb{R}_+$ define $K_a = \int P^t \, a(dt)$. The non-empty Borel set C is ν_a-small, if ν_a is a non-trivial measure with*

$$K_a(x, \cdot) \geq \nu_a(\cdot) \qquad \text{for all } x \in C.$$

If there is any such a and ν_a, we call C petite.

Lemma 6.2 *Suppose $C_{\min}$ is the set of germ-grain lists satisfying the following conditions:*

1. *each germ-grain pair covers at least one conditioning point;*
2. *if we remove one germ-grain pair from the list, then a conditioning point becomes uncovered.*

Then $C_{\min}$ is petite.

Proof: Consider the following germ-grain list

$$L_{\max} = \{(z_1, D), \ldots, (z_k, D)\}$$

where $\{z_i, i = 1, \ldots k\}$ are the conditioning points and D is a disc of radius 2. Each of the germ-grain pairs in $L_{\max}$ is given an independent unit mean Exponential life time. Assume we start the conditional birth-and-death process Φ in $L_{\max}$ and evolve it similarly to X, that is germ-grain pairs are born at rate $\mu()$ and have unit mean Exponential life times. At the end of a life time, a germ-grain pair dies or is perpetuated subject to the covering condition. Except for the starting configuration $L_{\max}$ we assume that any germ-grain pair which is born in Φ has a grain of distribution H. Let $Q^t(x, A) = \mathbb{P}(\Phi_t \in A | \Phi_0 = x)$. Fix positive T, $0 < \delta_1 < \delta_2 < T$, and set

$$A_i = \Big\{ \text{point } z_i \text{ is covered at time } T - \delta_1 \text{ by a germ-grain pair which was born in } [T - \delta_2, T - \delta_1] \text{ and whose grain has distribution } H \Big\}.$$

The FKG inequality [51, Theorem (2.4)] applies to the events $A_1, \ldots A_k$, since adding further germ-grain pairs increases the family of these events which are satisfied by a realization. Consequently we have

$$\mathbb{P}\Big(\bigcap_i A_i\Big) \geq \prod_i \mathbb{P}(A_i).$$

On the other hand we can find $\rho > 0$ such that $\mathbb{P}(A_i) > \rho$ for all i. It follows that $\mathbb{P}\Big(\bigcap_i A_i\Big)$ is positive. Now suppose that at time $T - \delta_1$ the conditioning points are all covered by $\ell \leq k$ newly born germ-grain pairs, so that $\bigcap_i A_i$ holds. By symmetry considerations and the memoryless property of the Exponential distribution we deduce that the probability of each of the ℓ new

germ-grain pairs lasting longer than all of the k germ-grain pairs in $L_{\max}$ is

$$\binom{l+k}{k}^{-1} \geq \binom{2k}{k}^{-1} > 0.$$

It follows that there is a positive probability that Φ_T is in a configuration which is E; thus the restriction of $Q^T(L_{\max}, \cdot)$ to $(E, \mathcal{E})$ is a non-trivial measure on $(E, \mathcal{E})$.

Recall that the grains of distribution H are almost surely contained in a disc of unit radius. As the coverage of any germ-grain pair in $L_{\max}$ is greater than that of any germ-grain pair from a configuration in $C_{\min}$, it follows that the probability that a germ-grain pair in $L_{\max}$ is allowed to die is smaller than that for a germ-grain pair from a configuration in $C_{\min}$. Thus if $P^t(x, A) = \mathbf{P}(X_t \in A | X_0 = x)$ and a the probability distribution which gives unit mass to T then it follows that for all $A \in \mathcal{E}$ and $x \in C_{\min}$

$$K_a(x, A) = P^T(x, A) \geq Q^T(L_{\max}, A).$$

Thus $C_{\min}$ is petite. □

We now show the following convergence result.

Theorem 6.3 *The process X is an ergodic Markov chain whose invariant distribution is the distribution of the target conditional Boolean model.*

Proof: We first show that X satisfies detailed balance with respect to the conditional Boolean model. Thus the target conditional Boolean model spec-

ifies the invariant distribution of X. It is sufficient to show that the germ-process of X satisfies detailed balance with respect to the germ distribution of the conditional Boolean model. Let φ denote a point configuration. Recall from Section 4.3.4 (27) the detailed balance condition for spatial birth-and-death processes:

$$b(\xi, \varphi)\, g(\varphi) \quad = \quad d(\xi, \varphi)\, g(\varphi \cup \{\xi\}) \quad \text{for } \xi \in W,$$

where g is the density of the equilibrium process with respect to a stationary unit rate Poisson process on the bounded window W. In our setting $b(\xi, \varphi) = \mu(\xi)$. Let $N^c(\varphi, G_\varphi)$ denote the number of conditioning points which are covered by the induced set $\cup_{\varphi_i \in \varphi}\ (\varphi_i \oplus G_i)$. Then the death rate of X is given by

$$d(\xi, \varphi) \quad = \quad I\Big(N^c(\varphi, G_\varphi) = N^c(\varphi \backslash \{\xi\}, G_{\varphi \backslash \{\xi\}})\Big),$$

where I denotes the indicator function. Now, the germ-process of the target conditional Boolean model has the density

$$f(\varphi) \propto \prod_{\varphi_i \in \varphi} \mu(\varphi_i)\, I\Big(N^c(\varphi, G_\varphi) = k\Big)$$

and so X satisfies detailed balance with respect to the target conditional Boolean model.

We proceed to show that X is positive Harris recurrent. According to [91, Theorem 4.3] the following condition is sufficient for positive Harris recurrence. Let $C = C_{\min}$ as defined in Lemma 6.2 and $\tau_C = \inf\{t \geq 0 : X_t \in C\}$,

then we require that

$$\mathbf{P}(\tau_C < \infty \mid X_0 = x) \quad = \quad 1 \qquad \text{for all } x \in E. \tag{46}$$

Let

$$T_1(x) \quad = \quad \min\{t \geq 0 : X_t \text{ covers all conditioning points} \mid X_0 = x\}.$$

For each conditioning point $z \in \mathcal{C}$, each time a germ-grain pair is born, there is a positive probability ϵ that this germ-grain pair covers z. As soon as z is covered, it cannot be uncovered again. Thus $T_1(x)$ is almost surely finite for any $x \in E$. Now let

$$T_2 \quad = \quad \min\{t \geq T_1(x) : \ Z_t = \emptyset\},$$

then T_2 is almost surely finite. As $\tau_c \leq T_2$ it follows that (46) is satisfied and thus that X is positive Harris recurrent.

To deduce ergodicity from positive Harris recurrence, we need to show that some skeleton chain $\check{X}$ of X is ϕ-irreducible, see [91, Theorem 6.1]. Let $\check{X}_n$ be the discrete-time, skeleton Markov chain with transition kernel P^T where $P^T(x, A) = \mathbf{P}(X_T \in A | X_0 = x)$. Let $\phi(A) = Q^T(L_{\max}, A)$ as defined in the previous lemma and let $n(x)$ be large enough such that

$$\mathbf{P}(X_{Tn(x)} \text{ covers all conditioning points } \mid X_0 = x) \quad = \quad \delta \quad > \quad 0.$$

Then if $\phi(A) = Q^T(L_{\max}, A) > 0$ and $n = n(x)$

$$\mathbf{P}(\check{X}_{n+1} \in A \mid \check{X}_0 = x) \quad = \quad P^{(n+1)T}(x, A) \ \geq \ P^{nT}(x, C_{min})\, Q^T(L_{\max}, A)$$

$$\geq \quad \delta \, \mathbf{P}(Z_{nT} = \emptyset) \, Q^T(L_{\max}, A) \quad > \quad 0.$$

Thus $\tilde{X}_n$ is ϕ-irreducible and hence X is an ergodic Markov chain. □

The CFTP algorithm for the conditional Boolean model will use the process Z together with the set of perpetuation times as an underlying process U. The process X can be realized as an adapted functional of U using Z_{-T} as the starting pattern at time $-T$. The process Z satisfies detailed balance with respect to the Boolean model Ψ, thus if we start Z in a realisation of Ψ, then Z is time-homogeneous.

We can only use U within a CFTP algorithm if we are able to extend U backwards in time. As Z is time-reversible, we can easily extend it backwards in time, but how about the perpetuation times? The time instances in an interval $[-T, 0]$ at which we assign perpetuation times and the total number of perpetuation times assigned during the interval crucially depend on how far in the past the process U is started. Thus extending the collection of perpetuation times backwards in time is not at all straight-forward. However, recall that we only have perpetuation if otherwise a conditioning point becomes uncovered. This link between perpetuation and conditioning points is the basis of the following procedure which solves the problem of backwards extension. To each point z_i in $\mathcal{C}$ we assign a Poisson process P_i of unit mean, supplying potential death times which will be used as follows. If a germ-grain pair of X is not allowed to die then the incident times of the

Poisson process assigned to the conditioning point covered by the germ-grain pair provide us with new potential death times. At the next of these death times the germ-grain pair becomes again a candidate for death. The properties of a unit rate Poisson process imply that the time till the next potential death time is Exponentially distributed with unit mean. In case of ambiguity, when the germ-grain pair covers more than one conditioning point, we need to decide which of the Poisson processes assigned to these conditioning points should provide the potential death times for the germ-grain pair. We only consider those Poisson processes which do not yet provide death times for a perpetuated germ-grain pair. It is arbitrary which of these processes we choose. There will always be at least one Poisson process to choose from because a germ-grain pair will only be perpetuated if it covers at least one conditioning point which is not covered by any other germ-grain pair. As we can simulate Poisson processes backwards in time, the underlying process U consisting of Z and the k Poisson processes $P_i, i = 1, \ldots, k$ can be extended backwards in time.

The following pseudo-code `MCMC-ConditionalBooleanModel` describes how we can produce a realization of X over the time-interval $[-T, 0]$ from a realization of U over the same time interval. Thus it describes explicitly how X can be realized as an adapted functional of U.

Notice that, although the algorithm starts X in the configuration of Z_{-T},

we could start X in any configuration whose non-perpetuated germ-grain pairs coincide with Z_{-T} and whose perpetuated germ-grain pairs are in $C_{\min}$. To each of the perpetuated germ-grain pairs we would assign a Poisson process according to the above rules.

The following definitions will be useful for the description of the algorithm. If the n^{th} unit-rate Poisson process $P_n, n = 1, \ldots, k$, supplies the potential death times of the perpetuated germ-grain pair (x, G), then the germ-grain pair is assigned the index n. This assignment is carried out by the algorithm $\texttt{index}_X(\cdot)$. The algorithm `BooleanModel` produces a realization of the process Z, the algorithm `Poisson` samples a unit mean Poisson process. The set I in the algorithm keeps track of the selection of Poisson processes which currently supply death times for perpetuated germ-grain pairs. Finally let $N^c(X_t)$ denote the number of conditioning points which are covered by the induced set of X_t.

```
MCMC-ConditionalBooleanModel(μ, H, C = {z_1, ..., z_k}, W × G × [−T, 0]):
   Z([−T, 0]) ← BooleanModel(μ, Leb(·), H, W × G × [−T, 0])
   for i = 1 to k
      P_j([−T, 0]) ← Poisson([−T, 0])
   {t_1, ..., t_{m(−T)}} ← ordered list of incident times of Z([−T, 0])
                           and P_j([−T, 0]) for all j = 1, ..., k
   X ← Z_{−T}
   I ← ∅
   for i = 1 to m(−T)

      if −t_i birth time of (x, G) then
         X ← X ∪ {(x, G)}

      if −t_i death time of (x, G) then
         if (N^c(X\{(x, G)}) = N^c(X)) then
            X ← X\{(x, G)}
         else
            I ← I ∪ {index_X ((x, G))}

      if −t_i incident time of P_j and j ∈ I then
         (x, G) ← grain with index j
         if (N^c(X\{(x, G)}) = N^c(X)) then
            X ← X\{(x, G)}
            I ← I\{j}
   return(X)
```

6.3 The Bounding Process

We now need to define a bounding process Y. To allow for an efficient representation of the bounding process, we check whether we can use monotone or anti-monotone CFTP. Thus we first need to define a partial order on E, the

space of finite germ-grain pair lists whose germs are in W and whose grains have distribution H.

Definition 6.4 *Consider two germ-grain pair lists:*
$L = \{(x_1, G_1), \ldots, (x_n, G_n)\}$ *and* $\tilde{L} = \{(\tilde{x}_1, \tilde{G}_1), \ldots, (\tilde{x}_m, \tilde{G}_m)\}$. *Let*

$$P(L) = \{j : (x_j, G_j) \in L \text{ is a perpetuated germ-grain pair }\}$$

be the index-set of the perpetuated germ-grain pairs of L. *We call*

$$L \preceq \tilde{L}$$

if there is at least one permutation $\pi = (\pi_1, \ldots, \pi_m)$ *of* $(1, \ldots, m)$ *such that we have*

(a) $(x_i, G_i) = (\tilde{x}_{\pi_i}, \tilde{G}_{\pi_i})$ *for all* $i \in \{1, \ldots, n\} \backslash P(L)$ *and*

(a) $x_i \oplus G_i \subseteq \tilde{x}_{\pi_i} \oplus \tilde{G}_{\pi_i}$ *for all* $i \in P(L)$.

In other words $L \preceq \tilde{L}$, if each non-perpetuated germ-grain pair in L is also in $\tilde{L}$ and each perpetuated germ-grain pair in L is covered by a germ-grain pair in $\tilde{L}$. As we will show later, this partial order allows for an efficient representation of the bounding process. Notice that the greater the multiple coverage of conditioning points by a germ-grain list, the smaller the chance of uncovering a conditioning point and thus the greater the chance of a permitted death. Hence the decision-rule for the acceptance of deaths is

anti-monotone with respect to the partial order $\preceq$ and so we may use anti-monotone CFTP.

In the following we define a minimal process $Y^{\min}$ and a maximal process $Y^{\max}$ as in Section 5.4.2. We use anti-monotone CFTP and apply the cross-over trick discussed in Section 5.4.2 by making deaths of the minimal process dependent on the current state of the maximal process and vice versa. The elements of the bounding process Y can be derived from:

$$Y_{-s}^{-T} = \left\{ L \in E : Y_{-s}^{\min(-T)} \preceq L \preceq Y_{-s}^{\max(-T)} \right\}.$$

We choose a time $-T < 0$ and define the evolution of the minimal and the maximal process in $[-T, 0]$ conditional on a realization of U over the same time interval. We give a rather careful specification of the algorithms using pseudo-code to be explicit about how the resultant coupling is defined.

The algorithm `BooleanInitial`(C, Z_{-T}) sets the initial configurations of the minimal and the maximal process as follows: The initial configuration of the minimal process is the list of germ-grain pairs describing the realization of Z at time $-T$. For the initial configuration of the maximal process we use the same list of germ-grain pairs but add an additional germ-grain pair (z, D) for each point $z \in C$, where D is a disc of radius 2. Recall that the grains in Z are almost surely contained in a disc of unit radius. Thus any germ-grain pair of Z covering some $z \in C$ will be covered by (z, D). If the realization of Z_{-T} yields N points, whose locations are $(x_1, \ldots, x_N)$ and whose marks are

$(G_1, \ldots, G_N)$, then the initial configuration of the minimal and the maximal process are given by

$$Y_{-T}^{\min(-T)} = \{(x_i,\ G_i) : i = 1, \ldots N\}$$
$$Y_{-T}^{\max(-T)} = Y_{-T}^{\min(-T)} \cup \{(z,\ D) : z \in \mathcal{C}\} = Y_{-T}^{\min(-T)} \cup L_{\max}, \quad (47)$$

where $L_{\max}$ is defined as in the proof of Lemma 6.2. Note that the initial configuration of the minimum process lies below the initial configuration of the maximal process with respect to the partial order $\preceq$. Furthermore, notice that the initial configuration of the maximal process does not lie in the state space E. This is motivated by the following. Assume we use the algorithm `MCMC-ConditionalBooleanModel` to produce a realization of X on some interval $[-T - S, 0]$ from a realization of U. Now condition on the realisation of U on $[-T, 0]$ and set the initial configurations of the minimal and the maximal process at time $-T$ as in (47). As the configuration of non-perpetuated germ-grain pairs of X a time $-T$ is equal to Z_{-T}, the configuration X_{-T} is bounded below by the configuration of the minimal process at time $-T$. Furthermore, although we do not know the exact configuration of perpetuated grains of X_{-T} (this requires knowledge of U prior to time $-T$), we know that X_{-T} will have at most k perpetuated germ-grain pairs. Moreover, the grain of each of these perpetuated germ-grain pairs has to lie within a disc of radius 2 centred at a conditioning point in $\mathcal{C}$, because only then can it possibly cover the conditioning point. It follows that the initial

configuration of the maximal process is an upper bound on the configuration of X_{-T}.

We call the additional germ-grain pairs $\{(z, D) : z \in \mathcal{C}\}$ in the maximal process *virtual* germ-grain pairs and treat these pairs as perpetuated. For $j = 1, \ldots, k$ we assign index j to the germ-grain pair (z_j, D), thus the potential death times of the perpetuated germ-grain pair (z_j, D) are provided by the j^{th} Poisson process of unit mean. The following pseudo-code describes `BooleanInitial`:

`BooleanInitial`$(\mathcal{C}, Z_{-T})$:
  $y^{min} \leftarrow \{(x, G) : (x, G) \in Z_{-T}\}$
  $y^{max} \leftarrow y^{min} \cup \{(z,\ D) : z \in \mathcal{C}\}$
  `for` $j = 1$ `to` $|\mathcal{C}|$
    `index`$(z_j, D) \leftarrow j$
  $I \leftarrow \{$ `index`$((z,\ D)) : z \in \mathcal{C}\}$
  `return`(y^{min}, y^{max}, I)

We can now give a recursive definition for the minimal and the maximal process on time interval $[-T, 0]$. Let $\mathcal{T} = \{-t_1, \ldots, -t_{m(-T)}\}$ be the combined list of the birth-and-death times of Z and the incident times of the k Poisson processes on $[-T, 0]$. Note that basic properties of Poisson point processes imply that none of the above times coincide. Thus a birth (death) time $-t$ specifies a unique germ-grain pair (x, G) with birth (death) time $-t$. Changes in the minimal and maximal process only occur at the times given

in $\mathcal{T}$. We extend the two processes in a right-continuous fashion as follows. Suppose that $Y_{-u}^{\min(-T)}$ and $Y_{-u}^{\max(-T)}$ are already defined and that $-t_i$ is the smallest element in $\mathcal{T}$ which is greater than $-u$. We set

$$\begin{aligned} Y_{-v}^{\min(-T)} &= Y_{-u}^{\min(-T)}, \\ Y_{-v}^{\max(-T)} &= Y_{-u}^{\max(-T)} \qquad \text{for } -u \le -v < -t_i. \end{aligned}$$

(If there is no such time, we set $Y_{-v}^{\min(-T)} = Y_{-u}^{\min(-T)}$ and $Y_{-v}^{\max(-T)} = Y_{-u}^{\max(-T)}$ for all $-v \le 0$.)

The changes in the configurations of the minimal and the maximal process at time $-t_i \in \mathcal{T}$ depend on whether $-t_i$ is a *birth* time of Z, a *death* time of Z or an *incident* time of one of the k Poisson processes.

Let $-t_i$ be a *birth* time. Then the algorithm `birth`$(-t_i,\ y^{\min}, y^{\max})$ adds the unique germ-grain pair (x, G) with birth time $-t_i$ to the current list by setting

$$\begin{aligned} Y_{-t_i}^{\min(-T)} &= Y_{-t_i-}^{\min(-T)} \cup \{(x, G)\}, \\ Y_{-t_i}^{\max(-T)} &= Y_{-t_i-}^{\max(-T)} \cup \{(x, G)\}, \end{aligned}$$

where $Y_{-t_i-}^{\min(-T)}$ denotes the state of the minimal process immediately prior to $-t_i$; an analogous notation is used for the maximal process.

```
birth(−t_i, y^min, y^max):
    (x, G) ← germ-grain pair with birth time −t_i
    y^min ← y^min ∪ {(x, G)}
    y^max ← y^max ∪ {(x, G)}
    return(y^min, y^max)
```

Note that a birth does not change the relation between the minimal and the maximal process with respect to $\preceq$. If the maximal process lies above the minimal process before a birth, it also does so afterwards.

Now let $-t_i$ be a *death* time of the unique germ-grain pair (x, G). Our action depends on the current state of the minimal and the maximal process. We remove the pair from the minimal process, if its removal from the maximal process does not uncover any conditioning points and vice versa for the maximal process. Thus we have

$$Y_{-t_i}^{\min(-T)} = \begin{cases} Y_{-t_i-}^{\min(-T)} \setminus \{(x,G)\} & \text{if } N^c\big(Y_{-t_i-}^{\max(-T)} \setminus \{(x,G)\}\big) \\ & \quad = N^c\big(Y_{-t_i-}^{\max(-T)}\big) \\ Y_{-t_i-}^{\min(-T)} & \text{otherwise} \end{cases}$$

$$Y_{-t_i}^{\max(-T)} = \begin{cases} Y_{-t_i-}^{\max(-T)} \setminus \{(x,G)\} & \text{if } N^c\big(Y_{-t_i-}^{\min(-T)} \setminus \{(x,G)\}\big) \\ & \quad = N^c\big(Y_{-t_i-}^{\min(-T)}\big) \\ Y_{-t_i-}^{\max(-T)} & \text{otherwise.} \end{cases}$$

The above task is performed by `death`$(-t_i,\ y^{\min}, y^{\max})$ as given below, which uses a global variable `death.max` to record whether a death was permitted

in the maximal process.

```
death(−t_i, y^min, y^max):
   (x, G) ← germ-grain pair with death time −t_i
   death.min ← FALSE
   death.max ← FALSE
   if (N^c(y^max \ {(x, G)}) = N^c(y^max)) then
      death.min ← TRUE
   if (N^c(y^min \ {(x, G)}) = N^c(y^min)) then
      death.max ← TRUE
   if death.min then
      y^min ← y^min \ {(x, G)})
   if death.max then
      y^max ← y^max \ {(x, G)})
   return(y^min, y^max, death.max)
```

Assume that the minimal process lies below the maximal process immediately before time $-t_i$. Recalling the definition of the partial order $\preceq$ it is clear that if the removal of a germ-grain pair from the minimal process does not uncover any conditioning point, then neither will its removal from the maximal process. On the other hand, if the removal from the maximal process uncovers a conditioning point, then its removal from the minimal process will also uncover at least one conditioning point. From this argument it follows that if a germ-grain pair dies in the maximal process then it dies in the minimal process. Similarly, if a germ-grain pair is perpetuated in the minimal process then it is perpetuated in the maximal process. In any

case the maximal process continues to lie above the minimal process at time $-t_i$.

If (x, G) is not removed in the maximal process, we assign an index to it which specifies the Poisson process which provides further potential death times for this germ-grain pair. The algorithm `perpetuate`$(-t_i,\ y^{\min}, y^{\max})$ chooses this index. For $j \in \{1, \ldots, k\}$ we call the Poisson process P_j *relevant* for (x, G), if $z_j \in C$ is covered by (x, G). A Poisson process which already provides death times for a perpetuated germ-grain pair is a *used* process. If we perpetuate germ-grain pair (x, G) in the maximal process then the index of the Poisson process chosen to do this is the smallest index of all Poisson processes which are relevant and not used in the maximal process. If the pair is perpetuated in the minimal process it is assigned the same index, thus it has the same potential death times as in the maximal process. Now, the algorithm must also account for the following scenario: a germ-grain pair (x, G) is perpetuated in the maximal process because its removal from the minimal process would uncover a conditioning point, but at the same time all conditioning points which are covered by (x, G) are already covered by perpetuated germ-grain pairs in the maximal process; Figure 19 illustrates this scenario. This means that there are no relevant Poisson processes which are not used in the maximal process. Let $I_{\min}$ be the index set of all relevant Poisson processes for (x, G) which are not used in the *minimal* process.

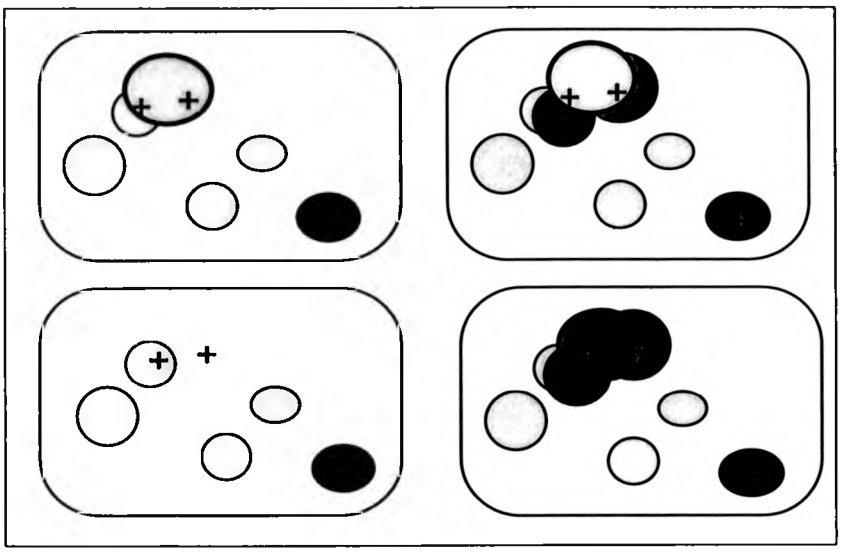

Figure 19: On the left hand side is the configuration of the minimal process, on the right hand side the configuration of the maximal process. The lighter shaded grains are non-perpetuated, the darker shaded grains are perpetuated. Suppose we have a potential death time for the thickly outlined grain. It dies in the minimal process, as the two conditioning points which it covers are covered by perpetuated grains in the maximal process. In the maximal process it is not allowed to die because removing it from the minimal process would uncover a conditioning point. The maximal process has already three perpetuated grains and thus no unused Poisson processes. We join the thickly outlined grain with the two perpetuated grains intersecting it.

In the above scenario each of the perpetuated germ-grain pairs which have an index in $I_{\min}$ is *enlarged by* (x, G). We say that a germ-grain (x_1, G_1) is enlarged by a germ-grain pair (x_2, G_2) if we replace it by the germ-grain pair (x_1, G), where G is such that $x_1 \oplus G = (x_1 \oplus G_1) \cup (x_2 \oplus G_2)$. The motivation for this procedure is to ensure that conditional on the underlying process U, the minimal and maximal process sandwich between them the realization of X regardless from how far in the past X is started, a more detailed motivation is given in the next Section 6.5.

Here is a summary of the algorithm `perpetuate`:

`perpetuate`$(-t_i,\ y^{\min}, y^{\max}, I)$:
  $I_r \leftarrow$ `index-set of relevant Poisson processes of` (x, G)
  $I_1 \leftarrow \cup\{i : i \in I_r$ `and` P_i `not used in` $y^{\max}\}$
  $I_2 \leftarrow \cup\{i : i \in I_r$ `and` P_i `not used in` $y^{\min}\}$
  `if` $(I_1 \neq \emptyset)$ `then`
    `index`$(x, G) \leftarrow \min\{i : i \in I_1\}$
    $I \leftarrow I \cup \{$`index`$(x, G)\}$
  `else`
    $y^{\max} \leftarrow y^{\max} \setminus \{(x, G)\}$
    `in` $y^{\max}$ `enlarge germ-grain pairs with index` $\in I_2$ `by` (x, G)
  `return`$(y^{\min}, y^{\max}, I)$

Finally let $-t_i$ be an *incident* time of the j^{th} Poisson process. We use the algorithm `perpet.death`$(-t_i,\ y^{\min}, y^{\max})$ to perform the following procedure. If there is no perpetuated germ-grain pair with index j in the maximal

and the minimal processes, we have no changes in the configuration of theses processes.

Suppose there is a perpetuated germ-grain pair (x, G) with index j both in the maximal process and in the minimal process. Then we delete the germ-grain pair from the maximal process if its removal from the minimal process does not uncover any conditioning point and vice versa for the minimal process. Thus we update the minimal and the maximal process as in the case when $-t_i$ is a death time.

If there is a perpetuated germ-grain pair (x, G) with index j in the maximal process which does not belong to the minimal process, then the minimal process does not change. We remove the pair from the maximal process, if adding the pair to the minimal process does not cover any additional conditioning point in the minimal process. We set

$$
\begin{aligned}
Y_{-t_i}^{\min(-T)} &= Y_{-t_i-}^{\min(-T)} \\
Y_{-t_i}^{\max(-T)} &= \begin{cases} Y_{-t_i-}^{\max(-T)} \setminus \{(x,G)\} & \text{if } N^c\big(Y_{-t_i-}^{\min(-T)} \cup \{(x,G)\}\big) \\ & \quad = N^c\big(Y_{-t_i-}^{\min(-T)}\big) \\ Y_{-t_i-}^{\max(-T)} & \text{otherwise.} \end{cases}
\end{aligned}
$$

Here is a summary:

```
perpet.death(−t_i, y^min, y^max, I):
   if j ∈ I then
      if (x, G) of index j ∈ y^min then
         pdeathmax = FALSE
```

```
        pdeathmin = FALSE
        if (N^c(y^max \ {(x,G)}) = N^c(y^max)) then
           pdeathmin = TRUE
        if (N^c(y^min \ {(x,G)}) = N^c(y^min)) then
           pdeathmax = TRUE
        if pdeathmin then
           y^min ← y^min \ {(x,G)}
        if pdeathmax then
           y^max ← y^max \ {(x,G)}
           I ← I \ {j}
     else
        if (N^c(y^min ∪ {(x,G)}) = N^c(y^min)) then
           y^max ← y^max \ {(x,G)}
           I ← I \ {j}
  return(y^min, y^max, I)
```

Combining all previous steps we obtain the algorithm BooleanEvo:

```
BooleanEvo(Z([−T,0]); P_j([−T,0]), j∈{1,...,k}; 𝒯; 𝒞):
  (y^min, y^max, I) ← BooleanInitial(𝒞, Z_{−T})
  for i = 1 to |𝒯|
     if (−t_i birth time) then
        (y^min, y^max) ← birth(−t_i, y^min, y^max)
     if (−t_i death time) then
        (y^min, y^max, death.max) ← death(−t_i, y^min, y^max)
        if (death.max = FALSE) then
           (y^min, y^max, I) ← perpetuate(−t_i, y^min, y^max, I)
     if (−t_i incident time of P_j) then
        (y^min, y^max, I) ← perpet.death(−t_i, y^min, y^max, I)
  return(y^min, y^max)
```

If $Y_0^{\max(-T)} = Y_0^{\min(-T)}$, then this configuration is a sample of the conditional Boolean model. If maximal and minimal process differ at time 0, we choose a new starting time $-S < -T$ and repeat above algorithm exchanging S for T. Recall that we have to extend backwards in such a way that we reuse the same realization of Z and of the k Poisson processes on $[-T, 0]$. The algorithms which carry out the extension backwards are called `Z-Extend` and `Pois-Extend`. We proceed to extend backwards in time until minimal and maximal process coalesce; if they coalesce we sample the configuration of the minimal process at time 0. The following pseudo-code provides a summary of the algorithm:

`CFTP-ConditionalBoolean`$(-T, \mathcal{C})$:
  $Z([-T,0]) \leftarrow$ `Z-Extend`$(\mu, -T)$
  `for` $i = 1$ `to` k
    $P_j([-T,0]) \leftarrow$ `Pois-Extend`$(-T)$
  $\mathcal{T} \leftarrow$ `ordered incident times of` $Z([-T,0])$, $P_j([-T,0]), j\in\{1,\ldots,k\}$
  $(y^{\min}, y^{\max}) \leftarrow$ `BooleanEvo`$(Z([-T,0]);\ P_j([-T,0]), j\in\{1,\ldots,k\};\ \mathcal{T};\ \mathcal{C})$
  `if` $y^{\min} = y^{\max}$ `then`
    `return`$(y^{\min})$
  `else`
    `CFTP-ConditionalBoolean`$(-2T, \mathcal{C})$

6.4 Properties of the CFTP Algorithm

The following relations are invariants over the course of the algorithm:

Lemma 6.5

$$Y_{-t}^{\min(-T)} \preceq Y_{-t}^{\max(-T)} \quad \textit{for any } -t \in [-T, 0]. \tag{48}$$

Proof: The relation (48) holds for the initial configuration of the minimal and the maximal process at time $-T$. Whenever a germ-grain pair dies in the maximal process it dies (subject to its existence) in the minimal process as long as $Y^{\min} \preceq Y^{\max}$ immediately before the time of death. Analogously, whenever a germ-grain pair is perpetuated in the minimal process it is also perpetuated in the maximal process. Thus the initial relation between the minimal and the maximal process is preserved. □

Lemma 6.6 *The maximal process always covers all conditioning points in* $\mathcal{C}$.

Proof: In the initial configuration of the maximal process all conditioning points are covered. Assume the removal of a germ-grain pair in the maximal process would uncover a conditioning point. Either this pair exists in the minimal process, in which case its removal from the minimal process will also uncover this conditioning point, or it does not exist in the minimal process, in which case the conditioning point is not covered in the minimal process. In neither case do we remove the pair from the maximal process. Thus a conditioning point is never uncovered in the maximal process. □

Lemma 6.7 *Let $-S < -T$, then for $-t \in [-T, 0]$*

$$Y_{-t}^{\min(-T)} \preceq Y_{-t}^{\min(-S)} \preceq Y_{-t}^{\max(-S)} \preceq Y_{-t}^{\max(-T)}. \qquad (49)$$

Proof: Note that the second inequality follows from Lemma 6.5. Recall that we derive $Y^{\min(-S)}$ and $Y^{\max(-S)}$ from a realization of Z on $[-S, 0]$ which is obtained by extending $Z([-T, 0])$ backwards in time. Thus for $-t \in [-T, 0]$, all four processes $Y_{-t}^{\min(-S)}$, $Y_{-t}^{\min(-T)}$, $Y_{-t}^{\max(-S)}$ and $Y_{-t}^{\max(-T)}$ are based on the same realization of Z. Furthermore note that for all four processes the times in $[-T, 0]$ when transitions might occur are identical.

We first show that the inequalities in (49) hold for $-t = -T$. What are the configurations of $Y_{-T}^{\min(-S)}$ and $Y_{-T}^{\max(-S)}$? Both configurations will contain all germ-grain pairs in Z_{-T}. Additionally they might contain some perpetuated germ-grain pairs. In $Y_{-T}^{\min(-S)}$ these perpetuated pairs are germ-grain pairs of $Z([-S, -T))$. In $Y_{-T}^{\max(-S)}$ the perpetuated pairs are either virtual germ-grain pairs, original germ-grain pairs of $Z([-S, -T))$ or enlarged germ-grain pairs of $Z([-S, -T))$. In either case any perpetuated germ-grain pair will have an induced set which lies within a disc of radius 2 centred at the appropriate conditioning point. Now let us compare these configurations with the initial configurations $Y_{-T}^{\min(-T)}$ and $Y_{-T}^{\min(-T)}$. The initial configuration of the minimal process $Y_{-T}^{\min(-T)}$ consists only of the germ-grain pairs in Z_{-T}

and does not have any perpetuated germ-grain pairs. Thus we have that

$$Y_{-T}^{\min(-T)} \preceq Y_{-T}^{\min(-S)},$$

which is the first inequality in (49) for time $-t = -T$. The initial configuration of the maximal process $Y_{-T}^{\max(-T)}$ consists of the germ-grain pairs in Z_{-T} plus a virtual germ-grain pair for each conditioning point. Recall that the induced set of any virtual germ-grain pair is a disc of radius 2 centred at the appropriate conditioning point for that perpetuated germ-grain pair. Thus any of the perpetuated germ-grain pairs in $Y_{-T}^{\max(-S)}$ is covered by virtual germ-grain pairs of $Y_{-T}^{\max(-T)}$. It follows that

$$Y_{-T}^{\max(-S)} \preceq Y_{-T}^{\max(-T)},$$

which is the third inequality in (49) for $-t = -T$. Note that for $-t \in [-T, 0]$ any germ-grain pair which is born in $Y_{-t}^{\min(-S)}$ and $Y_{-t}^{\max(-S)}$ is also born in $Y_{-t}^{\min(-T)}$ and $Y_{-t}^{\max(-T)}$ and vice versa. Now consider a potential death in the time interval $[-T, 0]$. Recall that the acceptance of a death for $Y_{-t}^{\max(-S)}$ depends on the current configuration $Y_{-t-}^{\min(-S)}$ of the minimal process immediately before time $-t$. Suppose $Y_{-t-}^{\min(-S)}$ lies above $Y_{-t-}^{\min(-T)}$. Then if the removal of the germ-grain pair from $Y_{-t-}^{\min(-T)}$ does not uncover a conditioning point it also does not do so in $Y_{-t-}^{\min(-S)}$. On the other hand if the removal of the germ-grain pair from $Y_{-t-}^{\min(-S)}$ uncovers a conditioning point it also does so in $Y_{-t-}^{\min(-T)}$. Therefore if the germ-grain pair dies in $Y_{-t}^{\max(-T)}$, it

also dies in $Y_{-t}^{\max(-S)}$, whereas if a germ-grain pair is perpetuated in $Y_{-t}^{\max(-S)}$ it is also perpetuated in $Y_{-t}^{\max(-T)}$. Now let us assume that $Y_{-t-}^{\max(-S)}$ lies below $Y_{-t-}^{\max(-T)}$. Then if we have a death in $Y_{-t}^{\min(-S)}$ we also have a death in $Y_{-t}^{\min(-T)}$ and if a germ-grain pair is perpetuated in $Y_{-t}^{\min(-T)}$ it is also perpetuated in $Y_{-t}^{\min(-S)}$. Thus the relation between the four processes is left unchanged by birth, death and perpetuation. Consequently the initial relation between the processes at time $-t = -T$ is preserved. Lemma 6.7 follows. □

6.5 The Target Markov Chain

In this section we construct an intermediate process Y^{mid} conditional on a realization of the minimal and the maximal process. We continue to prove that this intermediate process has the same stochastic properties as the target conditional spatial birth-and-death process X defined in the algorithm `MCMC-ConditionalBooleanModel`. Thus we show that, although our CFTP algorithm never simulates X directly, we can derive a realization of X from a realization of the minimal and the maximal process.

Suppose we have a realization of $Y_{-t}^{\max(-T)}$ and $Y_{-t}^{\min(-T)}$, $-t \in [-T, 0]$. The initial configuration of the intermediate process is the same as the initial configuration of the minimal process, that is we set

$$Y_{-T}^{\text{mid}(-T)} \quad = \quad Y_{-T}^{\min(-T)}.$$

Note that thus

$$Y_{-T}^{\min(-T)} \preceq Y_{-T}^{\mathrm{mid}(-T)} \preceq Y_{-T}^{\max(-T)}.$$

As is the case for the minimal and the maximal process, the intermediate process only changes at time instants given by the list $\mathcal{T} \subset [-T, 0]$ of birth-and-death times of Z and incident times of the k Poisson processes.

Suppose $-t_i \in \mathcal{T}$ is the *birth* time of germ-grain pair (x, G), then we add (x, G) to the intermediate process:

$$Y_{-t_i}^{\mathrm{mid}(-T)} = Y_{-t_i-}^{\mathrm{mid}(-T)} \cup \{(x, G)\}.$$

As this germ-grain pair is also added to the minimal and the maximal process, births preserve the initial relation between the three processes.

If $-t_i$ is the *death* time of the germ-grain pair (x, G), then we remove (x, G) from the intermediate process, if its removal does not uncover any conditioning point. We have

$$Y_{-t_i}^{\mathrm{mid}(-T)} = \begin{cases} Y_{-t_i-}^{\mathrm{mid}(-T)} \setminus \{(x, G)\} & \text{if } N^c\left(Y_{-t_i-}^{\mathrm{mid}(-T)} \setminus \{(x, G)\}\right) \\ & = N^c\left(Y_{-t_i-}^{\mathrm{mid}(-T)}\right) \\ Y_{-t_i-}^{\mathrm{mid}(-T)} & \text{otherwise.} \end{cases}$$

Assume

$$Y_{-t_i-}^{\min(-T)} \preceq Y_{-t_i-}^{\mathrm{mid}(-T)} \preceq Y_{-t_i-}^{\max(-T)}.$$

It follows that if the removal of (x, G) does not uncover a conditioning point in the minimal process then it also does not do so in the intermediate process. On the other hand if the removal of (x, G) uncovers a conditioning point in the intermediate process, then it also does so in the minimal process. Thus whenever (x, G) dies in the maximal process it also dies in the intermediate process; whenever it is perpetuated in the intermediate process, it is also perpetuated in the maximal process. Analogously, whenever (x, G) dies in the intermediate process it also dies in the minimal process; whenever it is perpetuated in the minimal process it is also perpetuated in the intermediate process. Hence the initial relation between the three processes is preserved.

If (x, G) is perpetuated then we need to assign an index to it which specifies the Poisson process used to supply the potential death times. We use the following rule. If there are any relevant Poisson processes for (x, G) which are not used in the maximal process at time t_i-, then we assign the smallest index of these processes to (x, G). This is the same index we assign to (x, G) in the maximal process. If there are no such Poisson processes then we use the smallest index of the relevant Poisson processes which are not used in the intermediate process. Note that if (x, G) is perpetuated, then there is at least one relevant Poisson process which is not yet used in the intermediate process. Recall that if there is no relevant Poisson process for (x, G) which is not used in the maximal process at time $-t_i-$, then we

use (x, G) to enlarge each of a set of perpetuated germ-grain pairs in the maximal process. This set comprises the pairs which have the index of a relevant Poisson process for (x, G) which is not used in the minimal process at time $-t_i-$. Now if $Y_{-t_i-}^{\min(-T)} \preceq Y_{-t_i-}^{\mathrm{mid}(-T)}$ then any Poisson process not used in the intermediate process is also not used in the minimal process. As a result the maximal process will contain a perpetuated germ-grain pair (x', G') which covers (x, G) and has the same potential death times.

We use the algorithm $\mathbf{index}_{\mathrm{mid}}$ to perform this task of assigning an index:

$\mathbf{index}_{\mathrm{mid}}((x, G),\ y^{\mathrm{mid}}, y^{\mathrm{max}}, I)$:

$I_r \leftarrow$ **index-set of relevant Poisson processes of** (x, G)

$I_1 \leftarrow \cup\{i : i \in I_r$ **and** P_i **not used in** $y^{\mathrm{max}}\}$

$I_2 \leftarrow \cup\{i : i \in I_r$ **and** P_i **not used in** $y^{\mathrm{mid}}\}$

if $(I_1 \neq \emptyset)$ **then**

  $\mathbf{index}(x, G) \leftarrow \min\{i : i \in I_1\}$

else

  $\mathbf{index}(x, G) \leftarrow \min\{i : i \in I_2\}$

$I \leftarrow I \cup \{\mathbf{index}(x, G)\}$

return(I)

Finally, if $-t_i$ is an *incident* time of the j^{th} Poisson process then we perform the following procedure. If there is no perpetuated germ-grain pair with index j in the intermediate processes, we have no change in this process.

Suppose there is a perpetuated germ-grain pair (x, G) with index j in the intermediate process. Then we delete the germ-grain pair if its removal from

the intermediate process does not uncover any conditioning point. We set

$$Y_{-t_i}^{\mathrm{mid}(-T)} = \begin{cases} Y_{-t_i-}^{\mathrm{mid}(-T)} \setminus \{(x,G)\} & \text{if } N^c\big(Y_{-t_i-}^{\mathrm{mid}(-T)} \setminus \{(x,G)\}\big) \\ & = N^c\big(Y_{-t_i-}^{\mathrm{mid}(-T)}\big) \\ Y_{-t_i-}^{\mathrm{mid}(-T)} & \text{otherwise} \end{cases}$$

Analogous to the case when $-t_i$ is a death time we find that if

$$Y_{-t_i-}^{\mathrm{min}(-T)} \preceq Y_{-t_i-}^{\mathrm{mid}(-T)} \preceq Y_{-t_i-}^{\mathrm{max}(-T)},$$

then

$$Y_{-t_i}^{\mathrm{min}(-T)} \preceq Y_{-t_i}^{\mathrm{mid}(-T)} \preceq Y_{-t_i}^{\mathrm{max}(-T)}.$$

The following pseudo-code summarizes how we obtain Y^{mid} on the time interval [-T,0]:

```
Intermediate(Z[−T,0]; P_j[−T,0], j∈{1,...,k}; y^max_[−T,0]; y^min_[−T,0]; 𝒯; C):
   y^mid ← y^min_{−T}
   I ← ∅
   for i = 1 to |𝒯|
      if −t_i birth time then
         (x,G) ← germ-grain with birth time −t_i
         y^mid ← y^mid ∪ {(x,G)}
      if −t_i death time then
         (x,G) ← germ-grain with death time −t_i
         if (N^c(y^mid) = N^c(y^mid \ {(x,G)})) then
            y^mid ← y^mid \ {(x,G)}
         else
            I ← I ∪ {index_mid((x,G), y^mid, y^max_{−t_i−}, I)}
      if −t_i incident time of P_j then
```

```
        if (x, G) ∈ y^mid and (x, G) has index j then
            if (N^c(y^mid) = N^c(y^mid\{(x, G)})) then
                y^mid ← y^mid\{(x, G)}
                I ← I\{j}
return(y^mid)
```

Figure 20 shows a sample of the maximal, the minimal and the intermediate process at a time before the three processes have coalesced. Note how the intermediate process is sandwiched between the minimal and the maximal process.

The intermediate process satisfies the following relations with the minimal and the maximal process.

Lemma 6.8 *Given the partial order $\preceq$, the intermediate process Y^{mid} continues to lie between the minimal and the maximal process as long as it does initially:*

$$Y_{-t}^{\min(-T)} \preceq Y_{-t}^{\mathrm{mid}(-T)} \preceq Y_{-t}^{\max(-T)} \quad \textit{for all } -t \in [-T, 0] \textit{ if}$$
$$Y_{-T}^{\min(-T)} \preceq Y_{-T}^{\mathrm{mid}(-T)} \preceq Y_{-T}^{\max(-T)}. \quad (50)$$

Proof: As argued in the description of the intermediate process if the intermediate process lies between the minimal and the maximal process immediately before a birth time, a death time or an incident time, then it will also do so immediately after this time. Thus the initial relation is preserved.

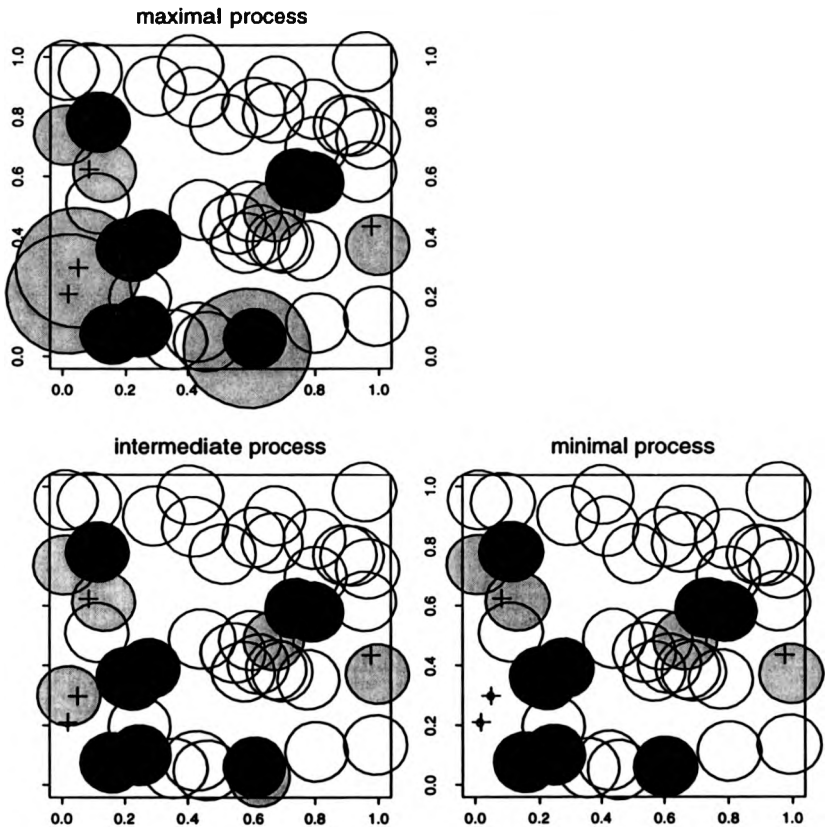

Figure 20: Illustration of maximal, minimal, and intermediate processes at a stage before coalescence has been achieved. The conditioning points are displayed as crosses. The germ intensity μ is 40 and the grains are discs of radius 0.1. The darker shaded grains are non-perpetuated grains which cover a conditioning point; the lighter shaded grains are perpetuated grains. Note that the maximal process contains virtual grains of radius 0.2. This figure is taken from [75].

Note that we make use of the fact that for each perpetuated germ-grain pair in the intermediate process there is a perpetuated germ-grain pair in the maximal process which covers the pair in the intermediate process and has the same potential death times. □

Lemma 6.9 *For all* $-t \in [-T, 0]$*:*

$$Y_{-t}^{\min(-T)} \preceq Y_{-t}^{\min(-S)} \preceq Y_{-t}^{\mathrm{mid}(-S)} \preceq Y_{-t}^{\max(-S)} \preceq Y_{-t}^{\max(-T)}. \qquad (51)$$

Proof: The lemma follows from Lemmas 6.7 and 6.8. □

We now show that Y^{mid} has the correct distribution when viewed on its own.

Lemma 6.10 *The process* $Y_{-t}^{\mathrm{mid}(-T)}$, $-t \in [-T, 0]$ *has the same stochastic properties as the conditional spatial birth-and-death process* X *produced by the algorithm* `MCMC-ConditionalBooleanModel` *and thus converges to the target conditional Boolean model.*

Proof: The initial configuration of Y^{mid} at time $-T$ is the list of germ-grain pairs in Z_{-T}. Whenever a germ-grain pair is born in Z it is also born in Y^{mid}, so Y^{mid} has the correct birth-rate. Whenever a germ-grain pair dies in Z, it is a candidate for death in Y^{mid}. It is then perpetuated if and only if its removal uncovers any conditioning point, and its perpetuation uses a unit rate Poisson process to discover a sequence of further candidate death

times, at each of which the germ-grain pair is perpetuated if and only if its removal uncovers any conditioning point. Thus the death rate is unit death rate per grain, except that transitions which uncover conditioning points are prohibited.

It follows that birth and death rates for Y^{mid} are the same as for the original X produced by the algorithm `MCMC-ConditionalBooleanModel`. In Theorem 6.3 we have shown that X converges to the conditional Boolean model and so the lemma is proved. □

6.6 Proofs of Requirements for CFTP

We will now show that the requirements of Theorem 5.2 are satisfied by our CFTP algorithm. We first show that U has the properties of an underlying process.

Lemma 6.11 *A Markov chain X whose equilibrium distribution is the conditional Boolean model, can be realized as an adapted functional of the underlying process U consisting of the process Z and k Poisson processes.*

Proof: The process X can be realized as an adapted functional of U using `MCMC-ConditionalBooleanModel`. The algorithm starts X in $g(U_{-T}) = Z_{-T}$. As Z is started in equilibrium it follows that $\{g(U_{-t}), -t \leq 0\}$ is time-homogeneous. Furthermore, we have shown in Theorem 6.3 that the process X converges to the conditional Boolean model. □

The following lemmas show that the bounding process which is defined through the minimal and the maximal process satisfies the necessary requirements.

Lemma 6.12 *The bounding process Y defined by*

$$Y_{-s}^{-T} \quad = \quad \left\{ L \in E : Y_{-s}^{\min(-T)} \preceq L \preceq Y_{-s}^{\max(-T)} \right\}.$$

is an adapted functional of U. Moreover, Y satisfies the funneling property:

$$Y_{-s}^{-T-u} \quad \subseteq \quad Y_{-s}^{-T} \qquad \text{for all } -u \leq 0 \text{ and } -T \leq -s \leq 0$$

and the sandwiching property:

$$X_{-s}^{-T}\left(Z_{-T}\right) \quad \in \quad Y_{-s}^{-T} \qquad \text{for all } -T \leq 0 \text{ and } -T \leq -s \leq 0.$$

Proof: The algorithm `BooleanEvo` developed in Section 6.3 shows that the minimal and the maximal process can be produced as adapted functionals of U. Dito follows for the bounding process. The funneling property follows from Lemma 6.8. For the sandwiching property recall that $Y_{-T}^{\mathrm{mid}(-T)} = Z_{-T}$ and furthermore that the intermediate process has the same stochastic properties as X. Therefore it is sufficient to show that Y^{mid} satisfies the sandwiching property. But this follows from Lemma 6.8 after observing that

$$Y_{-T}^{\min(-T)} \quad \preceq \quad Y_{-T}^{\mathrm{mid}(-T)} = Z_{-T} \quad \preceq \quad Y_{-T}^{\max(-T)}.$$

□

Lemma 6.13 *If the bounding process becomes a singleton it remains a singleton, that is if Y_{-s}^{-T} is a singleton, then Y_{-u}^{-T} is a singleton for all $-s \leq -u \leq 0$.*

Proof: If the configurations of the minimal and the maximal process coincide at time $-s$, then the acceptance rule for deaths in the minimal process is the same as in the maximal process. Hence if

$$Y_{-s}^{\min(-T)} = Y_{-s}^{\max(-T)} \text{ for some } -s \in [-T, 0],$$

then

$$Y_{-u}^{\min(-T)} = Y_{-u}^{\max(-T)} \text{ for all } -u \in [-s, 0].$$

□

Lemma 6.14 *The bounding process coalesces in almost surely finite time, that is*

$$T_C \quad = \quad \inf\{T \geq 0 : \ Y_0^{-T} \text{ is a singleton}\} \quad < \quad \infty \text{ almost surely.}$$

Proof: We need to show that minimal and maximal process coalesce almost surely. The proof follows a similar argument as in Lemma 6.2. Let $z_1, \ldots, z_k$ be the conditioning points. Fix positive t, δ, and set

$$A_i \quad = \quad \Big\{ \text{point } z_i \text{ is covered at time } t+\delta \text{ by a non-perpetuated germ-grain pair itself born in } [t, t+\delta] \Big\}.$$

The FKG inequality [51, Theorem (2.4)] applies to the events $A_1, \ldots A_k$ and so we have

$$\mathbb{P}\Big(\bigcap_i A_i\Big) \quad \geq \quad \prod_i \mathbb{P}(A_i)\,.$$

On the other hand we can find $\rho > 0$ such that $\mathbb{P}(A_i) > \rho$ for all i. It follows that $\mathbb{P}\Big(\bigcap_i A_i\Big)$ is positive.

Now suppose that at time $t + \delta$ the conditioning points are all covered by $\ell \leq k$ non-perpetuated germ-grain pairs, so that $\bigcap_i A_i$ holds. By symmetry considerations, the memoryless property of the Exponential distribution, and the fact that there can be at most k perpetuated grains covering $\mathcal{C} = \{z_1, \ldots, z_k\}$, we deduce that the probability of each of the ℓ non-perpetuated grains lasting longer than all of the $r \leq k$ perpetuated grains is

$$\binom{\ell + r}{\ell}^{-1} \quad \geq \quad \binom{2k}{k}^{-1} \quad > \quad 0\,.$$

It follows that there is a positive probability that at time $t + \delta + \epsilon$ for fixed positive ϵ the conditioning points are covered in the maximal process only by non-perpetuated grains: moreover the covering event may be refined so as to be independent of events prior to time t. Therefore a Borel-Cantelli argument shows that with probability one there is sufficiently large but finite t such that the conditioning points are covered in the maximal process only by non-perpetuated grains. But then there can be no perpetuated grains in the maximal process, and exactly then we may deduce that maximal and

minimal processes coalesce. □

The above work shows that the full algorithm does indeed produce a perfect sample of the conditional Boolean model. Figure 21 shows such a perfect sample. Nine conditioning points were placed in a unit square window. The underlying germ intensity of the Boolean model is $\mu = 40$ and the grains are discs of radius 0.1. The sample was produced on a computer (DEL 1486-DX2, 66 MHZ, 16 MB RAM) running MS-DOS in three and a half minutes.

6.7 Extensions and Conclusions

6.7.1 A Poisson Variable Approach

The developed CFTP algorithm not only allows us to sample a conditional Boolean model but also gives some information about the processes involved. Lantuéjoul drew our attention to the following example. Suppose we have a stationary Boolean model with intensity μ whose primary grains are discs of unit radius and which is conditioned to cover a single conditioning point. Then the number of grains which cover the conditioning point is a conditional Poisson variable of mean $\pi\mu$. In particular, the probability that the conditioning point is covered by exactly one grain is equal to

$$\mathbb{P}\big(N = 1 \,\big|\, N \geq 1\big) \quad = \quad \frac{\pi\mu}{\exp(\pi\mu) - 1},$$

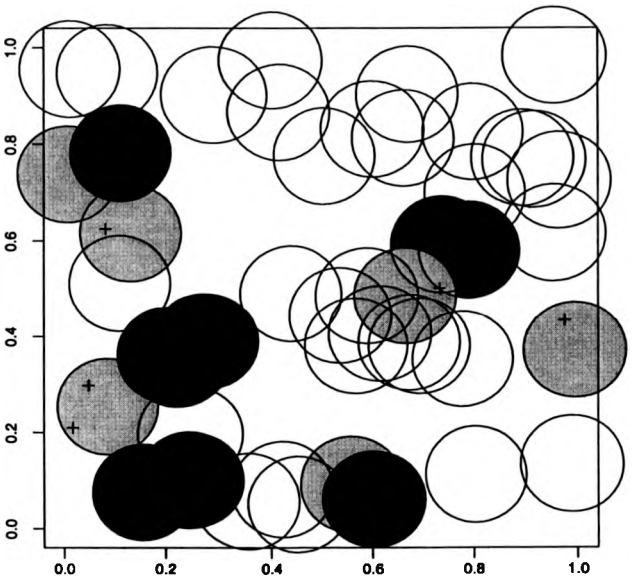

Figure 21: Perfect simulation of a conditional Boolean model with constant germ intensity $\mu = 40$ and grains of radius 0.1. The conditioning points are displayed as crosses. Non-perpetuated grains which cover conditioning points are shaded dark; the shading of perpetuated grains is lighter. This figure is taken from [75].

where N is a Poisson variable of mean $\pi\mu$. In our CFTP algorithm the event that we sample a Boolean random set in which the conditioning point is covered by a single germ-grain pair can occur in two cases. In the first case the configuration Z_0 does not cover the conditioning point, which occurs with probability $\exp(-\pi\mu)$. Then the CFTP algorithm ensures that the covering point is covered by a single perpetuated germ-grain pair. In the alternative case the configuration Z_0 covers the conditioning point with a single germ-grain pair, this happens with probability $\pi\mu\exp(-\pi\mu)$. Then the CFTP sample will cover the conditioning point with a single germ-grain pair if and only if it is not covered by a perpetuated germ-grain pair at time 0. Let $-S$ be the most recent incident time of the Poisson process supplying the perpetuation times and let N_{-t} denote the number of germ-grain pairs in Z_{-t} covering the conditioning point. There will be no perpetuated germ-grain pair at time 0 if N_{-t} does not hit 0 for $-t \in [-S, 0]$. Thus we have

$$\mathbb{P}\Big(N = 1 \;\Big|\; N \geq 1\Big) = \exp(-\pi\mu)\Big(1 + \pi\mu\,\mathbb{P}(N_{-t} \neq 0 \text{ for } -t \in [-S, 0])\Big),$$

and so

$$\mathbb{P}(N_{-t} \neq 0 \text{ for } -t \in [-S, 0]) \quad = \quad \frac{\pi\mu - 1 + \exp(-\pi\mu)}{\pi\mu(1 - \exp(-\pi\mu))}.$$

The above quantitity can be related to queuing theory.

Proposition 6.15 *If an $M/M/\infty$ queue (in equilibrium) is started in a busy period, then the probability that the busy period is longer than a unit mean*

Exponential waiting time is equal to

$$\mathbb{P}(N_{-t} \neq 0 \text{ for } -t \in [-S, 0]) \quad = \quad \frac{\pi\mu - 1 + \exp(-\pi\mu)}{\pi\mu(1 - \exp(-\pi\mu))}.$$

Previous to Lantuéjoul's remark, Kendall noted that the method described above for perfect simulation of conditional Boolean models can be abstracted to give a method for simulation of families of correlated Poisson random variables some of which are conditioned to be positive. This is currently being investigated by Kendall and Cai [17], we give a brief description of the set-up and point out its relation to the CFTP algorithm for the conditional Boolean model.

Assume we have a finite set $\mathcal{C} = \{x_1, \ldots, x_k\}$ of conditioning points. For each nonempty subset $A \subseteq \mathcal{C}$, we define an immigration-death process X_A of birth rate λ_A and unit death rate such that

$$\text{if} \quad B \subseteq A \quad \text{then} \quad X_A \subseteq X_B.$$

Here $X_A \subseteq X_B$ means that whenever there is an immigration in X_A then there is also one in X_B and whenever there is a death in X_A then X_B also suffers a death. The equilibrium distribution of any such process X_A is a Poisson process of mean λ_A.

If we consider a conditional Boolean model we can interpret the process X_A as the time-evolving number of germ-grain pairs whose grains cover *all* conditioning points in A.

Using the Möbius inversion formula [7] we can construct *independent* immigration-death processes Y_A of birth rate μ_A and unit death rate such that

$$X_B = \sum_{A:B\subseteq A} Y_A. \tag{52}$$

In the case of a conditional Boolean model the process Y_A describes the number of germ-grain pairs whose grains cover *all* conditioning points in A, but no conditioning point which is not in A. Note that some of the processes Y_A might have zero birth rate. In the conditional Boolean model this corresponds to a set A of conditioning points which cannot simultaneously be covered by a single germ-grain pair.

The equilibrium distribution of $(X_{\{x_1\}},\ldots,X_{\{x_k\}})$ is the *unconditional* correlated Poisson distribution. We can derive the conditional distribution through *perpetuation*. For each $A \subseteq S$ such that $\mu_A \neq 0$ we add an independent unit rate Poisson process Z_A to supply perpetuation times.

Now, we define the process $\tilde{Y}_A$ to have births in synchrony with Y_A, but special consideration is given to deaths which would lead to $\tilde{Y}_A = 0$. Namely, if such a death would cause $X_{\{x_i\}} = 0$ for some $x_i \in A$, we have perpetuation: the corresponding particle is allowed to survive till the next incident of Z_A. Thus perpetuation takes place if otherwise

$$\tilde{Y}_A(t-) = 1, \qquad \tilde{Y}_A(t) = 0$$

and there exists $x_i \in A$ such that

$$\tilde{Y}_B(t) \quad = \quad 0 \qquad \text{for all } B \neq A \text{ with } x_i \in B.$$

In the example of a conditional Boolean model with two conditioning points $z_1, z2$, say

$$\tilde{Y}_{\{z_1,z_2\}}(t-) \quad = \quad 1.$$

Then the germ-grain pair whose grain covers both x_1 and x_2 is allowed to die at time t if

$$\tilde{Y}_{\{z_1\}}(t) > 0 \quad \text{and} \quad \tilde{Y}_{\{z_2\}}(t) > 0.$$

In other words the germ-grain pair whose grain covers both conditioning points is allowed to die, if there is at least one germ-grain pair which covers z_1 (but not z_2) and at least one germ-grain pair which covers z_2 (but not z_1).

Now define

$$\tilde{X}_B \quad = \quad \sum_{A:B\subseteq A} \tilde{Y}_A.$$

Then the equilibrium distribution of $\tilde{X} = (\tilde{X}_{\{x_1\}}, \ldots, \tilde{X}_{\{x_k\}})$ is the correlated Poisson distribution conditioned to be positive.

Similarly to the conditional Boolean model we can now define a partial order on the state-space such that the acceptance rule for deaths is anti-monotone. We can then construct a dominated, anti-monotone CFTP algorithm which uses the independent immigration-death processes Y_A and Poisson processes $Z_A, A \subseteq C$ as an underlying process. A realisation of the

conditional Boolean model can be derived as follows. For each $A \subseteq \mathcal{C}$ we sample Y_A primary grains subject to the condition that the sampled grain is large enough to cover all conditioning points in A. For each sampled grain Ξ we then independently sample the associated germ uniformly on the area $\{y \in W : \ x_i \in (y \oplus \Xi) \ \forall x_i \in A\}$.

One difference between the Poisson random variable approach and the approach chosen for the conditional Boolean model is the number of Poisson processes supplying potential death times. For the Boolean model the number of Poisson processes was equal to k, in the setting here we can have up to $2^k - 1$ Poisson processes. The increased number of Poisson processes will lead to a computationally more expensive administration but on the other hand might reduce the time till coalescence of minimal and maximal process. It would be interesting to find conditions when the runtime reduces sufficiently to justify the increased administration cost.

Poisson variables are one-dimensional and thus the approach using correlated Poisson variables is simpler than the conditional Boolean model method. This simplification allowed Kendall and Cai [17] to verify their CFTP implementation for correlated Poisson variables by applying $\mathcal{X}^2$-tests to the output. As the CFTP algorithm for correlated Poisson variables is strongly related to the CFTP algorithm for conditional Boolean models, this also supports the confidence in the latter algorithm.

6.7.2 Further Extensions and Conclusions

Recall that we have conditioned the Boolean model to cover a set of points with grains. We could easily alter the condition to include the number of grains which cover a particular conditioning point. A more interesting condition would include connectivity constraints. This is a much harder problem for the following reason. In the case of the covering condition k Poisson processes are sufficient to supply all required perpetuation times. However, when imposing a connectivity constraint the number of required perpetuation times is not bounded above by the number of incident times of a fixed number of Poisson processes.

As a further extension we could consider a non-Poissonian process as an underlying germ process. In particular, we can combine the CFTP methods for locally finite point processes presented in [73, 74] with the perfect simulation method developed in this chapter and produce conditional germ-grain samples for locally finite germ processes.

The techniques used in the presented perfect simulation algorithm differ from the dominated CFTP methods developed for point processes in that we used perpetuation of life times whereas the latter use thinning of births.

More important is the novelty of implementing anti-monotone CFTP using a minimal and a maximal process on an *augmented* state space. This choice allowed for an efficient representation of the bounding process which

is crucial to the practicality of CFTP. Theorem 5.2 gives a lot of freedom when choosing the bounding process. This freedom can and should be exploited to produce efficient representations of bounding processes and thus practical CFTP algorithms.

7 Fill's Interruptible Algorithm

In Section 5.5.4 we described how an impatient user might introduce a subtle bias if using CFTP incorrectly and aborting long runs. We now present the perfect simulation algorithm developed by Fill [37] whose run time is independent of the state sampled. Thus a bias caused by aborting long runs cannot occur in this algorithm; the algorithm is interruptible.

In this chapter we first introduce Fill's interruptible algorithm in the finite state space setting for which it was orginally [37] developed. Besides being interruptible, Fill's algorithm is of great interest as it is an alternative perfect simulation method based on an approach different from CFTP. Although Fill's algorithm is usually presented as a version of rejection sampling, it originates from the concept of strong stationary times and duality developed by Diaconis and Fill [31]. In Section 7.2 we present the theory of strong stationary times and explain its relation to Fill's interruptible algorithm. Then, in Section 7.3, we develop an extension of Fill's algorithm to a point process example on a continuous state space. This section is a revised version of the presentation in Thönnes [140]. The final section reports on further extensions of Fill's algorithm.

7.1 The Algorithm for Finite State Spaces

Suppose we have a finite, partially ordered state space $(E, \leq)$ which possesses a minimal element $\hat{0}$ and a maximal element $\hat{1}$ such that $\hat{0} \leq x \leq \hat{1}$ for all $x \in E$.

Now, consider an ergodic Markov transition matrix P with stationary distribution π. We assume that $\pi(x) > 0$ for all $x \in E$. In accordance with the definition in (16) on page 67 we define the time-reversal $\tilde{P}$ of P as

$$\tilde{P}(x, y) \quad = \quad \frac{\pi(y)\, P(y, x)}{\pi(x)}, \qquad x, y \in E.$$

Let P^N denote the N-step transition matrix of P and $\tilde{P}^N$ the time-reversed N-step transition matrix.

Assume that the time-reversal transition matrix $\tilde{P}$ is monotone. Then, following Proposition 4.12, for any $x, y \in E$ with $x \leq y$ there exists an upward kernel $K_{(x,y)}$ with

$$\tilde{P}(y, y') \quad = \quad \sum_{x' \in E} \tilde{P}(x, x') K_{x,y}(x', y').$$

We require that we can simulate transitions from the measure $K_{(x,y)}(x', \cdot)$ whenever $x \leq y$ and $\tilde{P}(x, x') > 0$. If there is a monotone transition rule f for $\tilde{P}$ then Remark 4.13 on page 76 shows how to sample from $K_{(x,y)}(x', \cdot)$.

We can now define Fill's interruptible algorithm for the finite, monotone setting. Like any rejection sampling procedure Fill's algorithm proposes a sample which is then accepted or rejected with an appropriate probability.

The algorithm consists of three steps: the first step samples from the N-step transition matrix $P^N(\hat{0}, \cdot)$; the second and third step are designed such that this sample is accepted with the appropriate probability. Let X be a Markov chain with transition matrix P. Then the algorithm runs as follows:

(1) *Proposal step:* Start X in $\hat{0}$ and simulate it for N steps. Record the obtained trajectory $(X_0 = \hat{0}, \ldots, X_N = z)$. The state $X_N = z$ is the proposed sample state.

(2) *Reversal step:* Reverse the obtained trajectory in time leading to the trajectory

$$(\tilde{X}_0 = z,\ \tilde{X}_1, \ldots,\ \tilde{X}_N = \hat{0}) \quad = \quad (X_N = z,\ X_{N-1}, \ldots,\ X_0 = \hat{0}).$$

(3) *Rejection sampling step:* Together with the time-reversed trajectory use the upward kernels $K_{(x,y)}$ to simulate a second Markov chain $\tilde{Y}$ for N steps. The initial state of $\tilde{Y}$ is set as $\tilde{Y}_0 = \hat{1}$ and $\tilde{Y}_k$ is drawn from

$$K_{(\tilde{X}_{k-1}, \tilde{Y}_{k-1})}(\tilde{X}_k, \cdot) \qquad \text{for } k \in \{1, \ldots, N\}.$$

If $\tilde{Y}_N = \hat{0}$ then the proposed sample z is accepted, else it is rejected. If a proposed state is not accepted, the number of transitions N is doubled and the above three steps are repeated *independently* from the previous run.

We adopt the pseudocode notation to describe the algorithm. The algorithm `Markov1(`x`)` draws a sample from $P(x,\cdot)$ and `Markov2(`y,x,x'`)` draws a sample from $K_{(x,y)}(x',\cdot)$.

```
RejectionSampling(N):
   x_0 ← 0̂
   for k = 1 to N
      x_k ← Markov1(x_{k-1})
   for k = 0 to N
      x̃_k ← x_{N-k}
   ỹ_0 ← 1̂
   for k = 1 to N
      ỹ_k ← Markov2(ỹ_{k-1}, x̃_{k-1}, x̃_k)
   if (ỹ_N = 0̂)
      return(x_N)
   else
      RejectionSampling(2N)
```

Fill's algorithm draws a state z from $P^N(\hat{0},\cdot)$ and proposes it as a sample of $\pi(\cdot)$. The sample z must then be accepted with probability

$$\frac{1}{c}\,\frac{\pi(z)}{P^N(\hat{0},z)},$$

where c is an upper bound on the ratio

$$\frac{\pi(z)}{P^N(\hat{0},z)}.$$

According to the definition of $\tilde{P}$ and due to its monotonicity we have that

$$\frac{\pi(z)}{P^N(\hat{0},z)} \quad = \quad \frac{\pi(\hat{0})}{\tilde{P}^N(z,\hat{0})} \quad \leq \quad \frac{\pi(\hat{0})}{\tilde{P}^N(\hat{1},\hat{0})}.$$

Thus

$$c \quad = \quad \frac{\pi(\hat{0})}{\tilde{P}^N(\hat{1},\hat{0})}$$

is a valid choice for a rejection sampling procedure.

The following lemma [37, Lemma 7.4] shows that, because the upward kernels $K_{x,y}$ are used to produce $\tilde{Y}$, the probability of $\tilde{Y}_N = \hat{0}$ is equal to

$$\frac{1}{c}\,\frac{\pi(z)}{P^N(\hat{0},z)} \quad = \quad \frac{\tilde{P}^N(\hat{1},\hat{0})}{\pi(\hat{0})}\,\frac{\pi(\hat{0})}{\tilde{P}^N(z,\hat{0})} \quad = \quad \frac{\tilde{P}^N(\hat{1},\hat{0})}{\tilde{P}^N(z,\hat{0})},$$

whence the rejection sampling procedure is carried out correctly.

Lemma 7.1

$$\mathbb{P}\big(\tilde{Y}_N = \hat{0} \,\big|\, \tilde{X}_0 = z,\ \tilde{X}_N = \hat{0},\ \tilde{Y}_0 = \hat{1}\big) \quad = \quad \frac{\tilde{P}^N(\hat{1},\hat{0})}{\tilde{P}^N(z,\hat{0})}.$$

Proof: We follow the proof in Fill [37] filling in some details.

For now, assume we have a $\tilde{P}$-trajectory $(\tilde{X}_0 = \tilde{x}_0, \ldots, \tilde{X}_n = \tilde{x}_n)$ and we produce a trajectory $(\tilde{Y}_0 = \hat{1}, \ldots, \tilde{Y}_N)$ by sampling $\tilde{Y}_{k+1}$ from $K_{\tilde{x}_k,\tilde{Y}_k}(\tilde{x}_{k+1},\cdot)$ for $k = 0, \ldots, N-1$. Then

$$\mathbb{P}\big(\tilde{Y}_{k+1} = y', \tilde{X}_{k+1} = x' \,\big|\, \tilde{Y}_k = y, \tilde{X}_k = x\big) \quad = \quad \tilde{P}(x,x')K_{x,y}(x',y')$$

and so

$$\mathbb{P}\big(\tilde{Y}_{k+1} = y' \,\big|\, \tilde{Y}_k = y, \tilde{X}_k = x\big) \;=\; \sum_{x' \in E} \tilde{P}(x,x')K_{x,y}(x',y') \;=\; \tilde{P}(y,y').$$

Thus $\tilde{Y}_{k+1}$ is independent of $\tilde{X}_k$, and $\tilde{Y}$ is marginally a Markov chain with transition matrix $\tilde{P}$. Moreover, as $K_{(x,y)}$ is an upward kernel, $\tilde{Y}_{k+1} \geq \tilde{X}_{k+1}$ for all $k \in \{0, \ldots, N-1\}$ because $\tilde{Y}_0 \geq \tilde{X}_0$.

In Fill's algorithm the trajectory $(\tilde{X}_0 = z, \ldots, \tilde{X}_N = \hat{0})$ is produced by reversing a P-trajectory in time. But

$$\frac{\prod_{i=1}^{N} P(x_{i-1}, x_i)}{P^N(\hat{0}, z)} \quad = \quad \frac{\prod_{i=1}^{N} \tilde{P}(x_i, x_{i-1})}{\tilde{P}^N(z, \hat{0})}$$

and thus the distribution of the time-reversed trajectory is the same as of a $\tilde{P}$-trajectory which is conditioned to start in z and end in $\hat{0}$. Thus

$$\mathbb{P}\big(\tilde{Y}_N = \hat{0} \big| \tilde{X}_0 = z, \tilde{X}_N = \hat{0}, \tilde{Y}_0 = \hat{1}\big) = \frac{\tilde{P}^N(\hat{1}, \hat{0})}{\tilde{P}^N(z, \hat{0})},$$

where the dominator is due to the fact that we use a $\tilde{P}$-trajectory which is conditioned to start in z and end in $\hat{0}$. □

The unconditional probability of acceptance is

$$\frac{1}{c} \quad = \quad \frac{\tilde{P}^N(\hat{1}, \hat{0})}{\pi(\hat{0})} \quad = \quad \frac{P^N(\hat{0}, \hat{1})}{\pi(\hat{1})}$$

which due to the ergodicity of P converges to one as N tends to infinity. It follows that the algorithm terminates almost surely.

Here is a simple example which illustrates Fill's interruptible algorithm.

Example 7.2 (Simple Random Walk) *Consider the simple random walk X from Example 5.1. Recall that X lives on the state space $E = \{0, 1, 2\}$*

and can be simulated using the transition rule

$$f(j,U) = \begin{cases} \max\{j-1,0\} & \text{if } U = -1 \\ \min\{j+1,2\} & \text{if } U = 1, \end{cases}$$

where U is an independent Bernoulli random variable taking value 1 with probability p and value -1 with probability $q = 1-p$. The chain X is time-reversible and thus we can use f to sample from the time-reversal $\tilde{P}$ of P. Moreover, the transition rule f is monotone with respect to the natural ordering $\leq$, thus $\tilde{P}$ is monotone. The maximal element of the state space is 2, the minimal state is 0. To apply Fill's algorithm we need to sample from the upward kernel $K_{i,j}(i',\cdot)$ whenever $j \geq i$ and $\tilde{P}(i,i') > 0$. Remark 4.13 describes how we can sample from these upward kernels. Consider the conditional distribution of U given that $f(i,U) = i'$. Clearly, $U = -1$ if $i' = \max\{i-1,0\}$ and $U = +1$ if $i' = \min\{i+1,2\}$. Thus

$$K_{i,j}(i',j') = \begin{cases} I\big(j' = \max\{j-1,0\}\big) & \text{if } i' = \max\{i-1,0\} \\ I\big(j' = \min\{j+1,2\}\big) & \text{if } i' = \min\{i+1,2\}, \end{cases}$$

where I denotes the indicator function. Fill's interruptible algorithm now runs as follows. We start with some initial value for N, say $N = 2$ and perform the three steps of the algorithm.

(1) Proposal step: We sample two independent copies of U, say $U_1 = 1$ and $U_2 = -1$. We then start X in 0 and run it for two steps in accordance

with the sampled Bernoulli random variables. We thus produce the trajectory

$$(X_0,\ X_1,\ X_2) \quad = \quad (0,\ 1,\ 0)$$

and propose the state 0 *as a sample from* π.

(2) Reversal step: We now reverse the trajectory in time producing

$$(\tilde{X}_0,\ \tilde{X}_1,\ \tilde{X}_2) \quad = \quad (0,\ 1,\ 0).$$

(2) Rejection sampling step: The time-reversed trajectory specifies the upward kernels which we use to sample a trajectory of $\tilde{Y}$. *We set* $\tilde{Y}_0 = 2$ *and draw* $\tilde{Y}_{k+1}$ *from* $K_{(\tilde{X}_k, \tilde{Y}_k)}(\tilde{X}_{k+1}, \cdot)$ *for* $k = 0, 1$. *This produces the following trajectory*

$$(\tilde{Y}_0,\ \tilde{Y}_1,\ \tilde{Y}_2) \quad = \quad (2,\ 2,\ 1).$$

As $\tilde{Y}_2 \neq 0$ *we cannot accept* 0 *as a sample from* π. *We double the run time that is, we set* $N = 4$ *and independently repeat above three steps:*

(1) Proposal Step: We sample four independent copies of U, *say* $U_1 = -1$, $U_2 = 1$, $U_3 = 1$ *and* $U_4 = -1$. *We again start* X *in* 0 *and produce the trajectory*

$$(X_0,\ X_1,\ X_2,\ X_3,\ X_4) \quad = \quad (0,\ 0,\ 1,\ 2,\ 1).$$

Thus we propose the state 1 *as a sample from* π.

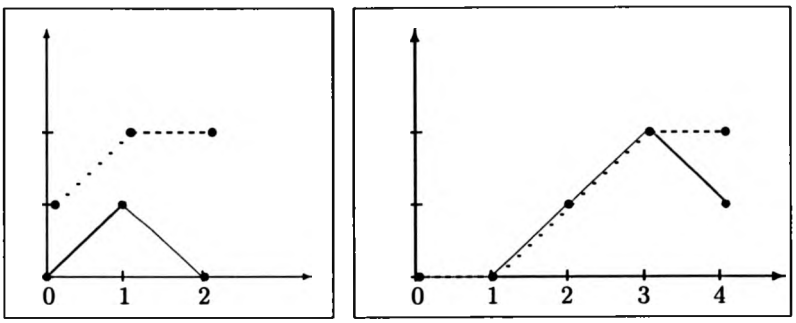

Figure 22: Illustration of Example 7.2. The left hand side shows the first iteration, the right hand side the second iteration of Fill's algorithm. X (gray lines) evolves from left to right along the time-axis, $\bar{Y}$ (dashed lines) evolves from the right to the left. The first iteration is not succesful as the path of $\bar{Y}$ does not meet the path of X at time 0. The second iteration is succesful, the path of $\bar{Y}$ meets the path of X at time 0.

---

(2) Reversal step: We reverse the trajectory in time producing

$$(\bar{X}_0,\ \bar{X}_1,\ \bar{X}_2,\ \bar{X}_3,\ \bar{X}_4) \quad = \quad (1,\ 2,\ 1,\ 0,\ 0).$$

(2) Rejection step: We set $\bar{Y}_0 = 2$ *and draw* $\bar{Y}_{k+1}$ *from the upward kernels specified by the time-reversed sequence, producing the trajectory*

$$(\bar{Y}_0,\ \bar{Y}_1,\ \bar{Y}_2,\ \bar{Y}_3,\ \bar{Y}_4) \quad = \quad (2,\ 2,\ 1,\ 0,\ 0).$$

This time $\bar{Y}_4 = 0$ *and thus we accept* 1 *as a sample from* π. *Figure 22 is an illustration of the above example.*

7.2 Strong Stationary Times and Strong Stationary Duality

In this section we discuss the concept of strong stationary times and strong stationary duality as developed by Diaconis and Fill [31] and summarize how Fill's interruptible algorithm relates to this theory, see also [37].

7.2.1 Strong Stationary Times

Let P be an ergodic transition matrix with stationary distribution π on a discrete state space E. Suppose X is a Markov chain on some probability space $(\Omega, \mathcal{F}, \mathbf{P})$ with initial distribution π_0 and transition matrix P .

Definition 7.3 (Strong Stationary Time) *Let T be a randomized stopping time for X. We call T a* strong stationary time, *if*

1. $X_T \sim \pi$ *and*
2. X_T *and* T *are independent.*

Above two conditions are equivalent to

$$\mathcal{L}(X_n \mid T = n) \quad = \quad \pi.$$

Definition 7.4 *Let $\pi_n(\cdot)$ be the distribution of X at time n. Then*

$$\text{sep}(n) \quad = \quad \text{sep}_{\pi_0}(n) \quad = \quad \max_{y \in E} \Big[1 - \frac{\pi_n(y)}{\pi(y)}\Big],$$

is the separation of X at time $n \in \mathbf{N}$.

Notice that $||\pi_n - \pi|| \leq \text{sep}(n)$.

Aldous and Diaconis [3, Proposition 3.2] show that

1. if T is a strong stationary time then

$$\text{sep}(n) \leq P(T > n), \qquad n \geq 0, \tag{53}$$

2. there exists a *minimal* strong stationary time $T_{\min}$ such that

$$\text{sep}(n) = P(T_{\min} > n), \qquad n \geq 0.$$

A minimal strong stationary time is also called a *time to stationarity*.

7.2.2 Strong Stationary Duality

We now construct a dual process X^* of X which is an absorbing process whose waiting time to absorption is a strong stationary time for X.

The basic set-up is as follows: Suppose we have an absorbing Markov chain X^* on the state-space E^* with initial distribution π_0^* and transition matrix P^* and furthermore, a function $\Lambda : E^* \times E \to [0,1]$. The aim is now to find a coupling (X^*, X) of X^* and X such that the function Λ is a *link*:

Definition 7.5 (Link) *The function $\Lambda : E^* \times E \to [0,1]$ is called a* link *for the chains X and X^* if*

$$\Lambda(x_n^*, \cdot) = \mathcal{L}(X_n \mid X_0^* = x_0^*, \ldots, X_n^* = x_n^*). \tag{54}$$

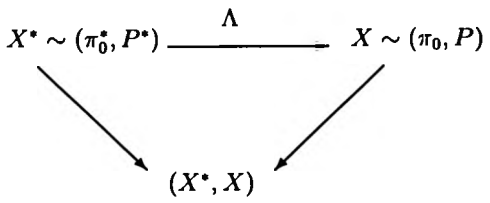

Figure 23: The basic set-up of strong stationary duality.

Example 7.6 (Set-valued dual) *Suppose X^* lives on the space of subsets of E and the Λ is defined as a truncation of the stationary distribution of π:*

$$\Lambda(x^*, x) = \begin{cases} \frac{\pi(x)}{H(x^*)} & \text{if } x \in x^* \\ 0 & \text{otherwise,} \end{cases}$$

where $H(x^*) = \sum_{x \in x^*} \pi(x)$. The dual X^* is then called a set-valued dual.

If we can find a coupling of X and X^* such that Λ is a link then we can show that X^* is a dual of X and moreover, we can construct a path of X^* given a path of X. Thus we can check whether X has reached stationarity by checking whether X^* has reached absorption.

We will first give a formal definition of a dual process and then state conditions on P^*, π_0^* and Λ such that we can find a coupling which satisfies (54). We then show how to construct this coupling and finally explain how to sample a path of X^* given a path of X.

Let $\coprod$ denote conditional independence, that is

$$X \coprod Y \mid Z \quad \text{if} \quad \mathcal{L}(X \mid Y, Z) = \mathcal{L}(X \mid Z).$$

The notation is due to Dawid [28] who studied the notion of conditional independence in a systematic fashion; [29] gives a formal treatment.

Definition 7.7 (Strong stationary dual) *Let $X^* = (X^*)_{n\geq 0}$ be a stochastic process on $(\Omega, \mathcal{F}, \mathbb{P})$ taking values in the discrete state space E^*. Assume that X^* has an absorbing state ∞. We call X^* a* strong stationary dual *for X if for all $n \geq 0$*

$$X \coprod X_n^* \mid X_0, \ldots, X_n, \quad \text{and} \tag{55}$$

$$\mathcal{L}(X_n \mid X_0^* = x_0^*, \ldots, X_{n-1}^* = x_{n-1}^*, X_n^* = \infty) = \pi. \tag{56}$$

Diaconis and Fill [31, Theorem 2.4] show the following result.

Theorem 7.8 (Strong stationary times and strong stationary dual) *(a) Let X^* be a strong stationary dual for X and T be the time to absorption in ∞ for X^*. Then T is a strong stationary time for X.*

(b) Let T be a strong stationary time for X and let $E^* = \mathbb{N}_0 \cup \{\infty\}$. Define

$$X_n^* = \begin{cases} n & \text{if } T > n, \\ \infty & \text{otherwise.} \end{cases}$$

Then X^* is a strong stationary dual for X and T is its waiting time until absorption.

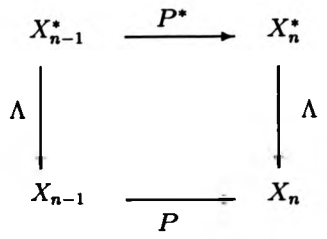

Figure 24: This illustration of the relation between X and its dual X^* is taken from [31].

Note that (b) is not the only way to construct a strong stationary dual, for any strong stationary time there are many strong stationary duals.

Before we construct the desired coupling, consider the following implications of the Definition 7.5 of a link. We have

$$\mathcal{L}(X_n \mid X^*_n = x^*) \quad = \quad \Lambda(x^*, \cdot) \tag{57}$$

$$\text{and} \qquad X_n \amalg X^*_{n-1} \mid X^*_n. \tag{58}$$

The conditional independence structure (55) of a strong stationary dual implies

$$X_n \amalg X^*_{n-1} \mid X_{n-1}. \tag{59}$$

The following theorem, which is [31, Theorem 2.17], states the conditions we need to impose on π^*_0, P^* and Λ to guarantee the existence of a coupling (X^*, X) which satisfies (57) – (59).

Theorem 7.9 *There exists a coupling (X^*, X) of X^* and X which is itself*

a Markov chain and satifies the equations (57) – (59) if and only if

$$\pi_0 = \pi_0^* \Lambda, \tag{60}$$

$$\Lambda P = P^* \Lambda. \tag{61}$$

(π_0^*, P^*) *is then called* algebraically dual *to* (π_0, P) *with respect to the link* Λ.

Note that if equation (61) holds and equations (57) – (59) are to be satisfied, the coupling needs to satisfy

$$\mathbf{P}\Big(X_{n-1} = x \;\Big|\; X_{n-1}^* = x^*, X_n = y\Big) = \Lambda(x^*, x)P(x, y)/\Delta(x^*, y) \tag{62}$$

$$\mathbf{P}\Big(X_n^* = y^* \;\Big|\; X_{n-1}^* = x^*, X_n = y\Big) = P^*(x^*, y^*)\Lambda(y^*, y)/\Delta(x^*, y), \tag{63}$$

where $\Delta = P^*\Lambda = \Lambda P$. For each $(x^*, y) \in E^* \times E$ let $p(y^*, x|x^*, y)$ be any distribution which has the marginals given in (62) and (63). We can easily construct such $p(y^*, x|y, x^*)$ by for example taking the product of the marginals given in (62) and (63). Assume that equations (60) and (61) hold, then we can use $p(y^*, x|y, x^*)$ to define a coupling which satisfies equations (57) – (59) as follows. The initial distribution is given by

$$\Pi_0(x^*, x) = \pi_0^*(x^*)\, \Lambda(x^*, x) \tag{64}$$

and the transition matrix is defined via

$$Q((x^*, x), (y^*, y)) = \frac{\Delta(x^*, y)\, p(y^*, x|x^*, y)}{\Lambda(x^*, x)}. \tag{65}$$

Now that we have found the desired coupling we can produce a path of X^* given a path of X such that the resulting bivariate path is a path of the Markov chain with initial distribution Π_0 and transition matrix Q:

1. Sample X_0^* according to $\mathcal{L}(X_0^* \mid X_0)$.
2. For $n \geq 1$, sample X_n^* from $\mathcal{L}\big(X_n^* \mid (X_0^*, X_0), \ldots, (X_{n-1}^*, X_{n-1}), X_n\big)$.

Definition 7.10 (Sharp dual) *The dual X^* of X is called* sharp *if its time to absorption is a minimal strong stationary time of X.*

Here is a simple condition [31, Remark 2.39] for the sharpness of a dual.

Lemma 7.11 *Assume we have a state $y \in E$ such that for all non-absorbing $y^* \in E^*$ either $\pi_n^*(y^*) = 0$ or else $\Lambda(y^*, y) = 0$. Then the dual X^* is sharp.*

Proof: As [31] omits a proof, we present the proof here. As $\pi_n = \pi_n^* \Lambda$ we have that

$$\pi_n(y) \quad = \quad \sum_{y^*} \pi_n^*(y^*) \Lambda(y^*, y) \quad = \quad \pi_n^*(\infty)\, \pi(y) \quad = \quad \pi(y) P(T^* \leq n),$$

where T^* is the random time till absorption for dual X^*. Then

$$\mathrm{sep}(n) \quad \geq \quad 1 - \frac{\pi_n(y)}{\pi(y)} \quad = \quad 1 - P(T^* \leq n)$$

and thus with (53) $\mathrm{sep}(n) = P(T^* > n)$, which means the dual is sharp.

7.2.3 The Algorithm and Strong Stationary Times

We return to the setting of Section 7.1 and follow [37, Section 9] in showing how Fill's interruptible algorithm is based on the theory of strong stationary times and dualtity. The agorithm essentially accepts a state drawn from $P^N(\hat{0}, \cdot)$ as a sample from π if a strong stationary dual X^* of X has reached the absorbing state at time N. It follows that the sample has the equilibrium distribution π.

We start by constructing a sharp set-valued dual X^* for the Markov chain X and then show how the dual X^* is used within the interruptible algorithm. Suppose $\tilde{f}(\cdot, U)$ is a monotone transition rule for the time reversal $\tilde{P}$ of the transition matrix P. Let E^* denote the collection of non-empty *order ideals* in the partially ordered state space $(E, \leq)$. A subset x^* of E is an order ideal in E if for any $x \in x^*$ and $y \leq x$ it follows that $y \in x^*$.

The chain X is started in the minimal state and thus has the initial distribution $\pi_0 = \delta_{\hat{0}}$, where δ_x puts unit mass on x. We start the dual X^* in $\{\hat{0}\}$ and thus set

$$\pi_0^* \quad = \quad \delta_{\{\hat{0}\}}.$$

We now define the link Λ. For $x^* \in E^*$ let $H(x^*) = \sum_{x \in x^*} \pi(x)$ then the link Λ is defined as

$$\Lambda(x^*, x) \quad = \quad \begin{cases} \pi(x)/H(x^*) & \text{if } x \in x^* \\ 0 & \text{otherwise.} \end{cases}$$

Now define

$$P^*(x^*, y^*) \quad = \quad \frac{H(y^*)}{H(x^*)} \, \mathbb{P}\Big(\bar{f}(y^*, U) = x^* \Big),$$

where $\bar{f}(y^*, U) = \{\bar{f}(y, U) : y \in y^*\}$.

Fill [37, Lemma 9.5] shows that P^* is a transition matrix on E^* and that (π_0^*, P^*) is algebraically dual to (π_0, P) with respect to the link Λ. Let X^* be the Markov chain with initial distribution π_0^* and transition matrix P^*. Then X^* has an absorbing state which is the order ideal E.

Fill continues to show in [37, Lemma 9.7] that the initial distribution $\Pi_0 = \delta_{(\hat{0}, \{\hat{0}\})}$ and the transition matrix

$$Q\Big((x^*, x), (y^*, y)\Big) \; = \; \frac{\pi(y)}{\pi(x)} \, \mathbb{P}\Big(\bar{f}(y, U) = x, \; \bar{f}(y^*, U) = x^*\Big)$$

defines a coupling (X^*, X) of X and X^* on the bivariate state space $\{(x^*, x) \in E^* \times E : x \in x^*\}$ which satisfies (57) – (61). Thus X^* is a strong stationary dual of X. Moreover, by applying Lemma 7.11 with $y = \hat{1}$, we can show that the dual X^* is sharp. As we have seen earlier we can now sample a trajectory of X^* given a trajectory of X such that the resulting bivariate trajectory is a trajectory of the coupling (X^*, X).

But how is this dual X^* used in the interruptible algorithm? In the proposal step we produce a (π_0, P)-trajectory

$$(X_0, \, X_1, \, \ldots, \, X_N) \quad = \quad (x_0 = \hat{0}, \, x_1, \, \ldots, \, x_N).$$

Given this trajectory we can simulate a (π_0^*, P^*)-trajectory such that the

resultant bivariate trajectory is a (Π_0, Q)-trajectory as follows. First observe that

$$\mathcal{L}\big(X_n^* \mid (X_0^*, X_0), \ldots, (X_{n-1}^*, X_{n-1}), X_n\big) \quad = \quad \frac{Q\big((X_{n-1}^*, X_{n-1}), (\cdot, X_n)\big)}{P(X_{n-1}, X_n)}.$$

Now, we set $X_0^* = \{\hat{0}\}$ and sample $X_n^* = x_n^*$ with probability

$$\begin{aligned} \frac{Q\big((x_{n-1}, x_{n-1}^*), (x_n, x_n^*)\big)}{P(x_{n-1}, x_n)} \quad &= \quad \frac{\mathbf{P}\big(\tilde{f}(x_n, U) = x_{n-1}, \tilde{f}(x_n^*, U) = x_{n-1}^*\big)}{\tilde{P}(x_n, x_{n-1})} \\ &= \quad \mathbf{P}\big(\tilde{f}(x_n^*, U) = x_{n-1}^* \,\big|\, \tilde{f}(x_n, U) = x_{n-1}\big) \end{aligned}$$

Suppose that for $n \in \{1, \ldots, N\}$, the random variable $\hat{U}_n$ has the distribution which is the conditional distribution of U given $\tilde{f}(x_n, U) = x_{n-1}$. Then

$$\frac{Q\big((x_{n-1}, x_{n-1}^*), (x_n, x_n^*)\big)}{P(x_{n-1}, x_n)} \quad = \quad \mathbf{P}\big(\tilde{f}(x_n^*, \hat{U}_n) = x_{n-1}^*\big).$$

Thus if $(\hat{U}_1, \ldots, \hat{U}_N) = (\hat{u}_1, \ldots, \hat{u}_N\}$ then

$$X_n^* \quad = \quad \{y \in E : \tilde{f}(y, \hat{u}_n) \in X_{n-1}^*\} \qquad \text{for } n \geq 1.$$

It follows that

$$X_n^* \quad = \quad \big\{y \in E : \tilde{f}(\cdots(\tilde{f}(\tilde{f}(y, \hat{u}_n), \hat{u}_{n-1})\cdots), \hat{u}_1) = \hat{0}\big\}.$$

Due to the monotonicity of f we have that X_n^* has reached absorption, that is $X_n^* = E$, if and only if $\hat{1} \in X_n^*$. Let

$$T \quad = \quad \min\{n \geq 1 : X_n^* = E\},$$

then T is a minimal strong stationary time of X.

Now, the rejection sampling step in Fill's algorithm determines whether $T \leq N$ by checking whether

$$\tilde{Y}_N \quad = \quad \tilde{f}(\cdots \tilde{f}(\tilde{f}(\hat{1}, \hat{u}_N), \hat{u}_{N-1}) \cdots), \hat{u}_1) \quad = \quad \hat{0}$$

and hence whether $X_N^* = E$. Thus Fill's algorithm is a very efficient procedure to check whether at time N the sharp set-valued dual X^* of X has yet reached its absorbing state. It follows that the algorithm samples X at a strong stationary time N. The obtained sample has the equilibrium distribution π and, furthermore, it is independent of N which makes the algorithm interruptible.

Remark 7.12 *Fill's algorithm doubles N after each iteration. The reason for this procedure is the following. If we would increase N by one in each iteration, the total number of forward transitions would be $\sum_{k=1}^{T} k$ which is of order T^2. The time-doubling procedure avoids this time-squaring increase in total number of transitions.*

To illustrate how the dual process X^* relates to Fill's algorithm consider again the simple random walk in Example 7.2.

Example 7.13 (Simple Random Walk). *Recall that X lives on the state*

space $E = \{0, 1, 2\}$ *and that the following function is a transition rule for* $\tilde{P}$*:*

$$\tilde{f}(j, U) \quad = \quad \begin{cases} \max\{j-1, 0\} & \text{if } U = -1 \\ \min\{j+1, 2\} & \text{if } U = 1, \end{cases}$$

where U *is an independent Bernoulli random variable with success probability* p*. We have already seen that* $U = -1$ *given* $f(i, U) = \max\{i-1, 0\}$ *and* $U = +1$ *given* $f(i, U) = \min\{i+1, 2\}$*. In the first iteration of the algorithm we produced the trajectory*

$$(X_0,\ X_1,\ X_2) \quad = \quad (0,\ 1,\ 0)$$

Thus

$$\hat{u}_1 \quad = \quad -1 \qquad \hat{u}_2 \quad = \quad +1$$

and so

$$X_1^* \ = \ \{i \in \{0,1,2\} : f(i, \hat{u}_1) = \max\{i-1, 0\} \in \{0\}\} \ = \ \{0, 1\} \quad \textit{and}$$

$$X_2^* \quad = \quad \{i \in \{0,1,2\} : f(i, \hat{u}_2) = \min\{i+1, 2\} \in \{0, 1\}\} \quad = \quad \{0\}.$$

As the dual process X^* *has not yet reached absorption, the algorithm correctly does not yet sample* X*. In the second iteration of the algorithm we sample the trajectory*

$$(X_0,\ X_1,\ X_2,\ X_3,\ X_4) \quad = \quad (0,\ 0,\ 1,\ 2,\ 1),$$

which leads to

$$(\hat{u}_1,\ \hat{u}_2,\ \hat{u}_3,\ \hat{u}_4) \quad = \quad (-1,\ -1,\ -1,\ +1).$$

Therefore

$$X_1^* = \{i \in \{0,1,2\} : f(i,\hat{u}_1) = \max\{i-1,0\} \in \{0\}\} = \{0,1\}$$

$$X_2^* = \{i \in \{0,1,2\} : f(i,\hat{u}_2) = \max\{i-1,0\} \in \{0,1\}\} = \{0,1,2\}$$

and thereafter

$$X_n^* = \{0,1,2\} \qquad \text{for } n \in \{3,4\}.$$

Thus in the second iteration we have sampled X at a strong stationary time and thus can accept the sample as a sample from π.

7.3 An Example from Point Process Theory

In this section we show how Fill's algorithm can be used to produce exact samples of the penetrable spheres mixture model and associated processes, thus generalizing Fill's algorithm to a continuous distribution. This section is based on [140].

Häggström, Van Lieshout and Møller [54] were the first to notice that the penetrable spheres mixture process can be sampled using a monotone Gibbs Sampler. They consequently developed a monotone CFTP algorithm for this setting. We use a similar partial order as in [54] and exploit the monotonicity of the Gibbs Sampler to define a perfect simulation method based on rejection sampling.

We begin by describing the penetrable spheres mixture model and its associated processes. We then develop a Fill-type perfect simulation method for this model and finally compare it with the analogous CFTP algorithm in [54].

7.3.1 The Penetrable Spheres Mixture Model

Let $W \subseteq \mathbb{R}^d$ be a bounded Borel set and let Ω denote the space of all finite point configurations in W without multiple points. The penetrable spheres mixture models model, see [56, 125, 126, 147], is a point process with two types of points, X and Y, on the product state space $\Omega \times \Omega$. The joint density of (X, Y) with respect to the product measure of two independent unit rate Poisson processes on W is given by

$$f(x, y) \quad = \quad \alpha_{\text{mix}} \, \beta_1^{n(x)} \, \beta_2^{n(y)} \, 1_{\{d(x,y)>R\}}, \tag{66}$$

where x and y are point configurations and $n(x)$ is the number of points in x. The factor α_{mix} is the normalising constant, $\beta_1, \beta_2 > 0$ are model parameters and $d(x, y)$ denotes the shortest distance between a point in x and a point in y. In the mixture model points of different type interact with each other via the constraint that points of different type must never be as close as R. Figure 25 on page 240 shows samples of the mixture model and illustrates how this constraint is satisfied.

The conditional distribution of X given Y is a homogeneous Poisson

process of intensity of β_1 on $W\backslash(Y \oplus B)$. Here B is a sphere of radius R and $\oplus$ denotes the Minkowski addition. Similarly, Y given X is a Poisson process of intensity β_2 on $W\backslash(X \oplus B)$. The mixture model is related to two other point processes, the area-interaction process, see (3) in section 2.2.2, and the continuum random-cluster model as described below.

The marginal distribution of X is an attractive area-interaction process [54, 147] whose density with respect to a unit-rate Poisson process on W is given by

$$p_1(x) \quad = \quad \alpha_1 \, \beta_1^{n(x)} \, \exp(\beta_2)^{-\lambda_L(x \oplus B)}.$$

Recall that λ_L denotes the Lebesgue measure on $\mathbf{R}^d$; the normalizing constant of the density $p_1(\cdot)$ is given by α_1. An analogous result holds for the marginal distribution of Y with the rôles of β_1 and β_2 exchanged.

The continuum random-cluster model [19, 77, 95] is specified by its density

$$p_2(z) \quad = \quad \alpha_2 \, \lambda_2^{n(z)} \, \gamma_2^{-c(z)} \tag{67}$$

with respect to a unit-rate Poisson process on W. Here α_2 is the normalizing constant and $\lambda_2, \gamma_2 > 0$ are model parameters. We use $c(z)$ to denote the number of connected components of $z \oplus G$, where G is a sphere of radius S. As shown in [54], for the *symmetric* penetrable spheres mixture process, that is when $\beta_1 = \beta_2$, the superposition of the two types of points $Z = X \cup Y$ is a continuum random-cluster model with radii $S = R/2$ and parameters

$\lambda_2 = \beta_1 = \beta_2$ and $\gamma_2 = 1/2$.

We mentioned earlier that the conditional distributions of X and Y are Poisson processes, which are easy to sample. Hence Gibbs sampling allows for an easy simulation of the mixture model, see [54, 95]. Given a sample of a symmetric mixture model, a sample of an attractive area-interaction model can be obtained by marginalising whereas a continuum random-cluster model with parameter $\gamma = 1/2$ can be obtained through superpositioning.

The following three sections define the three steps in our Fill-type algorithm: proposal, time-reversal and rejection sampling step.

7.3.2 The Proposal Step

In the first step of the rejection sampling algorithm we simulate a Markov chain, whose stationary distribution is the target distribution. Gibbs sampling is an easy way to obtain such a Markov chain for the penetrable spheres mixture model. We use the same Gibbs Sampler as [54]. Let (x_0, y_0) be an arbitrary point pattern in $\Omega \times \Omega$, then a single step of the Gibbs sampler is:

`GibbsSampling1`$((x_0, y_0))$:
  $y_1 \sim$ `Poisson`(β_2) `on` $W \backslash (x_0 \oplus B)$
  $x_1 \sim$ `Poisson`(β_1) `on` $W \backslash (y_1 \oplus B)$
  `return` $((x_1, y_1))$

Note that the Gibbs sampler does not actually use y_0 at all; it is included

in the function argument of `GibbsSampling1` for the sake of completeness. The equilibrium distribution of the resulting Markov chain does not depend on the order in which we update, hence we could also use:

`GibbsSampling2`$((x_0, y_0))$:
  $x_1 \sim$ `Poisson`(β_1) `on` $W \backslash (y_0 \oplus B)$
  $y_1 \sim$ `Poisson`(β_2) `on` $W \backslash (x_1 \oplus B)$
  `return` $((x_1, y_1))$

Here it is x_0 which is not being used.

In the rejection sampling algorithm we will iterate `GibbsSampling1` a fixed number of times and use the result as the proposal sample. Within one iteration of the Gibbs sampler we have two transition steps, that is the actual sequence of point patterns is as follows:

$$(x_0, y_0) \quad \longrightarrow \quad (x_0, y_1) \quad \longrightarrow \quad (x_1, y_1).$$

Let $p_x(x' \mid y)$ be the density of a Poisson process of intensity β_1 on $W \setminus [y \oplus B]$ and similarly, let $p_y(y' \mid x)$ be the density of a Poisson process of intensity β_2 on $W \setminus [x \oplus B]$. Then the first transition is made according to the transition kernel density $p_{y_0}(y \mid x_0)$ and the second transition is made according to $p_{x_0}(x \mid y_1)$. Detailed balance holds that is,

$$f(x, y)\, p_y(y' \mid x) \quad = \quad f(x, y')\, p_{y'}(y \mid x) \text{ and}$$

$$f(x, y)\, p_x(x' \mid y) \quad = \quad f(x', y)\, p_{x'}(x \mid y).$$

Lemma 7.14 *For both Gibbs Samplers* `GibbsSampling1` *and* `GibbsSampling2` *the whole state space is a small set and thus the Gibbs chain is positive Harris recurrent and moreover uniformly ergodic.*

We show the result for `GibbsSampling1`, the result follows analogously for `GibbsSampling2`. Assume without loss of generality that the window W has unit area. For any starting point (x_0, y_0) we have a positive probability of at least $\exp(-(\beta_1 + \beta_2))$ that one iteration of the Gibbs Sampler yields the empty point pattern $(x_1, y_1) = (\emptyset, \emptyset)$. Let g denote the distribution of a stationary unit-rate Poisson process. Then, if

$$P((x,y),B) \quad = \quad \int_B p_y(y'|x)p_x(x'|y')dg(x',y')$$

denotes the transition kernel of `GibbsSampling1`, then we have for all $(x, y) \in \Omega \times \Omega$

$$P^2((x,y),B) \;\geq\; \exp(-(\beta_1+\beta_2)) \int_B p_\emptyset(y'|\emptyset)p_\emptyset(x'|y')dg(x',y') \;=\; \nu(B).$$

It follows that the state space of the Gibbs chain is a small set. The lemma follows with Proposition 4.6. □

As pointed out earlier, Fill's idea requires that the transition kernels used have appropriate stochastic monotonicity properties. Consider the following partial order on $\Omega \times \Omega$ introduced in [54]:

Definition 7.15 *Let* $\Omega^2 = \Omega \times \Omega$. *We call*

$$(x,y) \;\leq_{\Omega^2}\; (x',y') \qquad \textit{if} \qquad x \subseteq x' \textit{ and } y \supseteq y'.$$

Both `GibbsSampling1` and `GibbsSampling2` are monotone with respect to the partial order in Definition 7.15 as we can define *monotone transition rules* for both Gibbs Samplers. Consider $f_1 : \Omega^2 \times \Omega \to \Omega^2$ and $f_2 : \Omega^2 \times \Omega \to \Omega^2$ such that

$$\begin{aligned} f_1((x,y), Z_1) &= (Z_1 \setminus [y \oplus B],\ y), \\ f_2((x,y), Z_2) &= (x,\ Z_2 \setminus [x \oplus B]), \end{aligned}$$

where Z_j is a Poisson point process on W with intensity $\beta_j, j = 1, 2$. These functions specify monotone transition rules since

$$\begin{aligned} \mathcal{L}\big(f_1(\ (x,y),\ Z_1)\ \big) &= p_x(\cdot \mid y) \\ \mathcal{L}\big(f_2(\ (x,y),\ Z_2)\ \big) &= p_y(\cdot \mid x). \end{aligned}$$

and furthermore if z_j is a realisation of Z_j then

$$f_j((x,y), z_j) \ \leq_{\Omega^2}\ f_j((x',y'), z_j) \ \text{ whenever } \ (x,y) \leq_{\Omega^2} (x',y'). \tag{68}$$

In the following we use the above transition rules whenever we perform `GibbsSampling1` or `GibbsSampling2`.

In contrast to Fill's setting, the Markov chain here is defined on an uncountable state space which, although it has a partial order $\leq_{\Omega^2}$, does not have a unique maximal or minimal state with respect to this partial order. Instead Häggström et al. [54] call an element $(x,y) \in \Omega^2$ *quasimaximal*, if

$$x \oplus B \supseteq W \qquad \text{and} \qquad y = \emptyset.$$

Similarly, the element $(x, y) \in \Omega^2$ is *quasiminimal*, if

$$x = \emptyset \qquad \text{and} \qquad y \oplus B \supseteq W.$$

Although these states are not unique maximal and minimal states, Häggström et al. [54] show that they serve as effectively maximal and minimal states. If we use the transition rules f_1 and f_2 to couple transitions, then after one iteration of the sequential Gibbs Sampler a path started from a quasimaximal state will lie above the path started from any other state. Similarly after one iteration of the sequential Gibbs Sampler a path started from a quasiminimal state will lie below the path started from any other state. Thus the concept can be used to define a monotone CFTP algorithm.

In the following we equip a suitable projection of the state space Ω^2 with a slightly partial order $\preceq$. This allows us to define elements which are (albeit not unique) maximal or minimal in the usual sense.

Definition 7.16 (Partial Order) *We call*

$$(x, y) \preceq (x', y') \qquad \textit{if} \qquad x \oplus B \subseteq x' \oplus B \quad \textit{and} \quad y \oplus B \supseteq y' \oplus B.$$

Now, suppose $(x, y) \preceq (x'y')$ then

$$f_j\big((x, y), z_j\big) \preceq f_j\big((x', y'), z_j\big) \qquad \text{for } j = 1, 2$$

and thus both `GibbsSampling1` and `GibbsSampling2` are monotone with respect to $\preceq$. Furthermore, an element $(x, y) \in \Omega^2$ is maximal, if

$$x \oplus B \supseteq W \qquad \text{and} \qquad y \oplus B = \emptyset.$$

Similarly the state $(x, y) \in \Omega^2$ is minimal, if

$$x \oplus B = \emptyset \qquad \text{and} \qquad y \oplus B \supseteq W.$$

Note that a $\preceq$-minimal element is $\leq_{\Omega^2}$-quasiminimal and a $\preceq$-maximal element is $\leq_{\Omega^2}$-quasimaximal.

In the proposal step of our algorithm we start in a minimal state and run N iterations of `GibbsSampling1`. The time-reversal step then reverses the obtained trajectory; the next section examines the properties of the time-reversed trajectory.

7.3.3 The Time-Reversal Step

Suppose we run N iterations of `GibbsSampling1` and thus obtain a realisation of the sequence $(X_k, Y_k)_{k=0}^N$. In the time-reversal step of our algorithm we then reverse the obtained sequence of point patterns in time that is, we set

$$\tilde{X}_k = X_{N-k} \qquad \text{and} \qquad \tilde{Y}_k = Y_{N-k}.$$

In this section we will prove a result about the distribution of this time-reversed sequence, which will be essential when showing that the developed algorithm samples the correct target distribution. Fill uses an analogous result but without proof, see Section 7.3 in [37]. We include a proof, as the result is unfamiliar in the context of point processes. Let $f(x, y)$ denote the density of the mixture model given in (66).

Definition 7.17 *Let $\tilde{p}_x(x'|y)$ and $\tilde{p}_y(y'|x)$ be the time-reversal of the transition kernel densities $p_x(x'|y)$ and $p_y(y'|x)$, that is*

$$\tilde{p}_x(x'|y) = \frac{f(x',y)p_{x'}(x|y)}{f(x,y)} \quad \text{and} \quad \tilde{p}_y(y'|x) = \frac{f(x,y')p_{y'}(y|x)}{f(x,y)} \tag{69}$$

for arbitrary x, y such that $f(x,y) > 0$.

Note that an easy calculation shows that $\tilde{p}_x(x'|y) = p_x(x'|y)$ and $\tilde{p}_y(y'|x) = p_y(y'|x)$: in this case the components of the Gibbs Sampler are *time-reversible*, which is not generally required for Fill's method. However it should be noticed that the Gibbs Sampler chain itself is not reversible: `GibbsSampling1` and `GibbsSampling2` stand in duality.

Let the time-reversed N-step transition density be given by

$$\tilde{p}^N\Big((\tilde{x}_N, \tilde{y}_N) \,\Big|\, (\tilde{x}_0, \tilde{y}_0)\Big) = \int \tilde{p}_{\tilde{x}_0}(\tilde{x}_1|\tilde{y}_0)\, \tilde{p}_{\tilde{y}_0}(\tilde{y}_1|\tilde{x}_1) \times \cdots \times \tilde{p}_{\tilde{x}_{N-1}}(\tilde{x}_N|\tilde{y}_{N-1}) \times \tilde{p}_{\tilde{y}_{N-1}}(\tilde{y}_N|\tilde{x}_N) dg(\tilde{x}_1, \tilde{y}_1) \cdots dg(\tilde{x}_{N-1}, \tilde{y}_{N-1}). \tag{70}$$

Lemma 7.18 *Let $(X_k, Y_k)_0^N$ be the chain produced by N iterations of* `GibbsSampling1` *started in (X_0, Y_0). We reverse the sequence in time and obtain the sequence $(\tilde{X}_k, \tilde{Y}_k)_0^N$. Then, conditional on $(\tilde{X}_0, \tilde{Y}_0) = (q,r)$ and $(\tilde{X}_N, \tilde{Y}_N) = (s,t)$, the time-reversed sequence $(\tilde{X}_k, \tilde{Y}_k)_0^N$ has the density*

$$\Big[\tilde{p}^N\Big((s,t) \,\Big|\, (q,r)\Big)\Big]^{-1} \tilde{p}_q(\tilde{x}_1|r)\, \tilde{p}_r(\tilde{y}_1|\tilde{x}_1) \cdots \tilde{p}_{\tilde{x}_{N-1}}(s|\tilde{y}_{N-1})\, \tilde{p}_{\tilde{y}_{N-1}}(t|s). \tag{71}$$

In other words, given the state at time 0 is (q,r) and the state at time N is (s,t), the time-reversed sequence is distributed like a sequence produced by

the time-reversal kernel densities and conditioned to start in (q, r) at time 0 and to end in (s, t) at time N. Note that the time-reversed N-step transition density specifies the distribution of the output produced by N iterations of `GibbsSampling2`.

Proof: Let $h_j(\cdot, \cdot)$ be the density of (X_j, Y_j). The time-reversed sequence $(\tilde{X}_k, \tilde{Y}_k)_0^N$ has the same density as the original sequence which is

$$\begin{aligned} & h^N\Big((x_n, y_n)_0^N \Big) \\ &= h_0(x_0, y_0) p_{y_0}(y_1|x_0) p_{x_0}(x_1|y_1) \cdots p_{y_{N-1}}(y_N|x_{N-1}) p_{x_{N-1}}(x_N|y_N). \end{aligned}$$

Substituting the time-reversal kernel densities for the transition kernel densities this density becomes

$$\begin{aligned} h^N\Big((x_n, y_n)_0^N \Big) &= \frac{h_0(x_0, y_0) f(x_N, y_N)}{f(x_0, y_0)} \tilde{p}_{y_1}(y_0|x_0) \tilde{p}_{x_1}(x_0|y_1) \tilde{p}_{y_2}(y_1|x_1) \\ &\qquad \times \cdots \times \tilde{p}_{y_N}(y_{N-1}|x_{N-1}) \tilde{p}_{x_N}(x_{N-1}|y_N). \end{aligned}$$

Substituting $\tilde{x}_k$ and $\tilde{y}_k$ for x_{N-k} and y_{N-k} and reordering leads to

$$\begin{aligned} h^N\Big((\tilde{x}_n, \tilde{y}_n)_0^N \Big) &= \frac{h_0(\tilde{x}_N, \tilde{y}_N) f(\tilde{x}_0, \tilde{y}_0)}{f(\tilde{x}_N, \tilde{y}_N)} \tilde{p}_{\tilde{x}_0}(\tilde{x}_1|\tilde{y}_0) \tilde{p}_{\tilde{y}_0}(\tilde{y}_1|\tilde{x}_1) \tilde{p}_{\tilde{x}_1}(\tilde{x}_2|\tilde{y}_1) \\ &\qquad \times \cdots \times \tilde{p}_{\tilde{x}_{N-1}}(\tilde{x}_N|\tilde{y}_{N-1}) \tilde{p}_{\tilde{y}_{N-1}}(\tilde{y}_N|\tilde{x}_N). \end{aligned}$$

If we now condition on $(\tilde{X}_0, \tilde{Y}_0) = (q, r)$ and $(\tilde{X}_N, \tilde{Y}_N) = (s, t)$, the conditional density of the time-reversed sequence is

$$\begin{aligned} & h^N\Big((\tilde{x}_n, \tilde{y}_n)_0^N \;\Big|\; (\tilde{X}_0, \tilde{Y}_0) = (q, r), (\tilde{X}_N, \tilde{Y}_N) = (s, t) \Big) \\ &\quad = \tilde{p}^N\Big((s, t) \;\Big|\; (q, r) \Big)^{-1} \tilde{p}_q(\tilde{x}_1|r) \tilde{p}_r(\tilde{y}_1|\tilde{x}_1) \cdots \tilde{p}_{\tilde{x}_{N-1}}(s|\tilde{y}_{N-1}) \tilde{p}_{\tilde{y}_{N-1}}(t|s). \quad (72) \end{aligned}$$

□

We make some further statements, which we will need when we prove that our algorithm produces exact samples of the mixture model.

Let $p^N\big((x_N, y_N)\big|(x_0, y_0)\big)$ be the N-step transition density of `GibbsSampling1`, that is

$$p^N\Big((x_N, y_N)\ \Big|\ (x_0, y_0)\Big) = \int p_{y_0}(y_1|x_0)\, p_{x_0}(x_1|y_1) \times \cdots \times p_{y_{N-1}}(y_N|x_{N-1}) \times p_{x_{N-1}}(x_N|y_N) dg(x_1, y_1) \cdots dg(x_{N-1}, y_{N-1}).$$

Due to detailed balance we have that

$$f(x_0, y_0)\, p^N\Big((x_N, y_N)\ \Big|\ (x_0, y_0)\Big) = f(x_N, y_N)\, \bar{p}^N\Big((x_0, y_0)\Big|(x_N, y_N)\Big). \quad (73)$$

Now let $z_{\hat{0}}$ be a minimal pattern and $z_{\hat{1}}$ be a maximal pattern, then as a result of the monotonicity of `GibbsSampling1` and `GibbsSampling2`

$$\begin{aligned} p^N\big(z_{\hat{1}}\big|z_{\hat{0}}\big) &\leq p^N\big(z_{\hat{1}}\big|(x_0, y_0)\big) \qquad \text{for any } (x_0, y_0) \succeq z_{\hat{0}} \text{ and} \\ \bar{p}^N\big(z_{\hat{0}}\big|z_{\hat{1}}\big) &\leq \bar{p}^N\big(z_{\hat{0}}\big|(\tilde{x}_0, \tilde{y}_0)\big) \qquad \text{for any } (\tilde{x}_0, \tilde{y}_0) \preceq z_{\hat{1}}. \end{aligned} \quad (74)$$

After obtaining the time-reversed trajectory of our Markov chain, we simulate a second Markov chain, whose final state will be used as criterion for accepting or rejecting the proposed sample. The next section constructs this second Markov chain.

7.3.4 The Rejection Sampling Step

We now describe the Markov chain on which we base the decision whether to accept or reject the proposed sample. The evolution of this second Markov chain is dependent on the evolution of the time-reversed trajectory. Suppose we have the following transitions in the time-reversed trajectory

$$(\tilde{x}_0, \tilde{y}_0) \quad \longrightarrow \quad (\tilde{x}_1, \tilde{y}_0) \quad \rightarrow \quad (\tilde{x}_1, \tilde{y}_1).$$

Let $(\tilde{v}_0, \tilde{w}_0)$ be a starting pattern of our second Markov chain, such that $\tilde{w}_0 \oplus B \subseteq \tilde{y}_0 \oplus B$. One iteration of the acceptance algorithm is then defined as follows:

`Acceptance`$((\tilde{v}_0, \tilde{w}_0), (\tilde{x}_0, \tilde{y}_0), (\tilde{x}_1, \tilde{y}_1))$:
  $\tilde{r}_1 \sim$ `Poisson`(β_1) `on` $(\tilde{y}_0 \oplus B) \cap W$
  $\tilde{v}_1 \leftarrow [\, \tilde{x}_1 \cup \tilde{r}_1 \,] \setminus [\, \tilde{w}_0 \oplus B \,]$
  $\tilde{w}_1 \leftarrow \tilde{y}_1 \setminus [\, \tilde{v}_1 \oplus B \,]$
  `return` $(\tilde{v}_1, \tilde{w}_1)$

Note that this algorithm does not make use of the point pattern $\tilde{v}_0$. Also note that $\tilde{w}_1 \oplus B \subseteq \tilde{y}_1 \oplus B$, so that we can apply the above algorithm iteratively.

We designed the acceptance algorithm such that the first transition is made according to the conditional distribution of the $f_1((\tilde{v}_0, \tilde{w}_0), Z_1)$ given $f_1((\tilde{x}_0, \tilde{y}_0), Z_1) = (\tilde{x}_1, \tilde{y}_0)$ and the second according to the conditional distribution of the transition rule $f_2((\tilde{v}_1, \tilde{w}_0), Z_2)$ given $f_2((\tilde{x}_1, \tilde{y}_0), Z_2) = (\tilde{x}_1, \tilde{y}_1)$.

Hence the transitions of our second chain are made according to the upward kernel density of the transition kernel defined by $\bar{p}_x(x'|y)$ and $\bar{p}_y(y'|x)$ as in Proposition 4.12; see Remark 4.13.

The acceptance algorithm was chosen such that the rejection sampling algorithm has the appropriate acceptance probability. To show this we will need the following result about the distribution of the output produced by N iterations of `Acceptance`. Suppose the sequence $(\bar{X}_k, \bar{Y}_k)_0^N$ was produced by running N iterations of `GibbsSampling1` and reversing the obtained trajectory. Let (q, r) be the state at time 0 and (s, t) be the state at time N of this time-reversed sequence. Using this sequence we run N iterations of the acceptance algorithm and produce the sequence $(\bar{V}_k, \bar{W}_k)_0^N$. The distribution of $(\bar{V}_N, \bar{W}_N)$ is given as follows, in a simple consequence of Lemma 7.18.

Corollary 7.19 *Let $(\bar{V}_0, \bar{W}_0) = (u_1, u_2)$ be a valid starting pattern for the acceptance algorithm and let $(\bar{V}_N, \bar{W}_N)$ be the output produced by N iterations of* `Acceptance`. *Conditional on $(\bar{X}_0, \bar{Y}_0) = (q, r)$ and $(\bar{X}_N, \bar{Y}_N) = (s, t)$, $(\bar{V}_N, \bar{W}_N)$ is distributed according to*

$$\frac{\bar{p}^N\Big((\bar{v}_N, \bar{w}_N) \;\Big|\; (u_1, u_2)\Big)}{\bar{p}^N\Big((s, t) \;\Big|\; (q, r)\Big)}.$$

Proof: Assume that the transitions in the time-reversed sequence are made according to the time reversal kernel densities $\bar{p}_x(x'|y)$ and $\bar{p}_y(y'|x)$. As the transition kernel densities coincide with their time-reversals, we have that

1. $\tilde{X}_{k+1}$ is a Poisson point process of intensity β_1 on $W \setminus [\, \tilde{Y}_k \oplus B \,]$ and

2. $\tilde{Y}_{k+1}$ is a Poisson point process of intensity β_2 on $W \setminus [\, \tilde{X}_{k+1} \oplus B \,]$.

The acceptance algorithm chooses $\tilde{R}_{k+1}$ as a Poisson point process of intensity β_1 on $(\tilde{Y}_k \oplus B) \cap W$. Due to the independence properties of the Poissson point process it follows that $\tilde{V}_{k+1}$ is a Poisson point process of intensity β_1 on $W \setminus [\, \tilde{W}_k \oplus B \,]$. Now, as

$$\tilde{W}_k \oplus B \quad \subseteq \quad \tilde{Y}_k \oplus B, \qquad \text{we have} \qquad \tilde{X}_{k+1} \oplus B \quad \subseteq \quad \tilde{V}_{k+1} \oplus B$$

and so

$$\tilde{W}_{k+1} \quad = \quad \tilde{Y}_{k+1} \setminus \tilde{V}_{k+1}$$

is a Poisson point process of intensity β_2 on $W \setminus [\, \tilde{V}_{k+1} \oplus B \,]$. In other words, the transitions

$$(\tilde{V}_k, \tilde{W}_k) \quad \longrightarrow \quad (\tilde{V}_{k+1}, \tilde{W}_k)$$

are distributed according to $\tilde{p}_{\tilde{V}_k}(\tilde{v} \mid \tilde{W}_k)$ and the transitions

$$(\tilde{V}_{k+1}, \tilde{W}_k) \quad \longrightarrow \quad (\tilde{V}_{k+1}, \tilde{W}_{k+1})$$

according to $\tilde{p}_{\tilde{W}_k}(\tilde{w} \mid \tilde{V}_{k+1})$. As shown in Lemma 7.18, conditional on $(\tilde{X}_0, \tilde{Y}_0) = (q, r)$ and $(\tilde{X}_N, \tilde{Y}_N) = (s, t)$ the transitions of the time-reversed sequence are made according to the time-reversal kernel densities. Hence conditional on (q, r) and (s, t) the N-step transition kernel density of the

acceptance algorithm is given by

$$\frac{\tilde{p}^N\Big((\tilde{v}_N, \tilde{w}_N) \,\Big|\, (u_1, u_2)\Big)}{\tilde{p}^N\Big((s,t) \,\Big|\, (q,r)\Big)},$$

where the denominator is due to our condition. □

7.3.5 The Rejection Sampling Algorithm

Following Fill's program we combine the algorithms of Sections 7.3.2 – 7.3.4 to a rejection sampling algorithm as follows. We first determine a minimal pattern $(z_1, z_2) = z_{\hat{0}}$, that is we set $z_1 = \emptyset$ and choose z_2 deterministically such that $z_2 \oplus B \supseteq W$. We will use $z_{\hat{0}}$ as the starting pattern for the proposal algorithm. We then set $z_{\hat{1}} = (z_2, z_1)$, which leads to a maximal pattern. This will be the starting pattern for the acceptance algorithm. Note that as $z_1 = \emptyset$, the pattern $z_{\hat{1}}$ is always a valid starting pattern for the acceptance algorithm. Throughout the following we keep the above patterns fixed. Note that the rejection sampling algorithm will not actually depend on the choice of z_2 as the proposal algorithm only makes use of the first component of the starting pattern and the acceptance algorithm only of the second one, see the remarks made when describing the algorithms. The rejection sampling algorithm is then defined as follows:

```
RejectionSampling(N):
   (x_0, y_0) ← z_0̂
   for k = 1 to N
```

```
   (x_k, y_k) ← GibbsSampling1(x_{k-1}, y_{k-1})
for k = 0 to N
   (x̃_k, ỹ_k) ← (x_{N-k}, y_{N-k})
(ṽ_0, w̃_0) ← z_î
for k = 1 to N
   (ṽ_k, w̃_k) ← Acceptance((ṽ_{k-1}, w̃_{k-1}), (x̃_{k-1}, ỹ_{k-1}), (x̃_k, ỹ_k))
if ((ṽ_N, w̃_N) = z_ĵ)
   return((x_N, y_N))
else
   RejectionSampling(2N)
```

Note that if a sample is not accepted, the following run of the Markov chain is independent of the previous runs.

If we use conventional MCMC, we just iterate `GibbsSampling1` (say) N times and assume that the produced sample has the stationary distribution. The reversal and acceptance step in our algorithm ensures that the sample actually has the stationary distribution. As in the finite case the algorithm is a form of rejection sampling: the proposal algorithm proposes a sample from the N-step transition density $p^N(\cdot \mid z_{\hat{\jmath}})$ and the acceptance algorithm accepts it with an appropriate probability. Having obtained a sample (x, y) from $p^N(\cdot \mid z_{\hat{\jmath}})$, we wish to accept it as a sample from f with probability $f(x,y) \setminus (c\, p^N((x,y) \mid z_{\hat{\jmath}}))$, where c is an upper bound on the ratio $f(x,y) \setminus p^N((x,y) \mid z_{\hat{\jmath}})$. Due to detailed balance (73) and the monotonicity of the Gibbs sampler (74)

$$\frac{f(x,y)}{p^N((x,y)|z_{\hat{\jmath}})} \quad = \quad \frac{f(z_{\hat{\jmath}})}{\tilde{p}^N(z_{\hat{\jmath}}|(x,y))} \quad \leq \quad \frac{f(z_{\hat{\jmath}})}{\tilde{p}^N(z_{\hat{\jmath}}|z_{\hat{\imath}})}.$$

So we can use $c = f(z_{\hat{0}}) \setminus \bar{p}^N(z_{\hat{0}}|z_{\hat{1}})$ and hence we want to accept (x, y) with probability

$$\frac{f(x,y)}{p^N((x,y)|z_{\hat{0}})} \times \frac{1}{c} = \frac{f(z_{\hat{0}})}{\bar{p}^N(z_{\hat{0}}|(x,y))} \times \frac{\bar{p}^N(z_{\hat{0}}|z_{\hat{1}})}{f(z_{\hat{0}})} = \frac{\bar{p}^N(z_{\hat{0}}|z_{\hat{1}})}{\bar{p}^N(z_{\hat{0}}|(x,y))}.$$

But the acceptance algorithm accepts the sample exactly with this probability as the following lemma shows:

Lemma 7.20 *The probability that the acceptance algorithm accepts a proposed sample (x, y) that is*

$$\mathbf{P}\Big((\bar{v}_N, \bar{w}_N) = z_{\hat{0}} \;\Big|\; (\bar{x}_0, \bar{y}_0) = (x, y),\ (\bar{x}_N, \bar{y}_N) = z_{\hat{0}},\ (v_0, w_0) = z_{\hat{1}}\Big)$$

is equal to

$$\frac{\bar{p}^N(z_{\hat{0}}|z_{\hat{1}})}{\bar{p}^N(z_{\hat{0}}|(x,y))}.$$

Proof: This follows directly from Corollary 7.19 using $(\bar{x}_0, \bar{y}_0) = (x, y)$, $(\bar{x}_n, \bar{y}_n) = z_{\hat{0}}$, $(\bar{v}_0, \bar{w}_0) = z_{\hat{1}}$ and $(\bar{v}_N, \bar{w}_N) = z_{\hat{0}}$. □

Lemma 7.21 *The number of iterations I (which will be random) needed by the rejection sampling algorithm to produce output is finite with probability one.*

Proof: The unconditional probability of acceptance is

$$c^{-1} = \frac{\bar{p}^N(z_{\hat{0}}|z_{\hat{1}})}{f(z_{\hat{0}})} = \frac{p^N(z_{\hat{1}}|z_{\hat{0}})}{f(z_{\hat{1}})}.$$

Due to ergodicity the above expression converges to one as $N \to \infty$. Therefore the probability of output at the j^{th} iteration given no prior output converges to one for $j \to \infty$. □

7.3.6 Simulation

We implemented the rejection sampling algorithm in a C program to demonstrate that the exact sampling of the penetrable spheres model with this algorithm is feasible in practice. Figure 25 shows some samples of the penetrable spheres mixture model obtained through the new algorithm. We use the same display as Häggström et al. in [54]. The first type of points is pictured as a dot, the second type is pictured as a cross. The circles show the area around a point in which no point of the other type is allowed. Note that for a larger value of β phase transitions seem to begin to occur and one of the components dominates. The implementation of our algorithm also enables us to compare it to the analogous algorithm using CFTP in [54] with respect to memory and expected number of transitions needed.

Fill [37] shows that for a finite Markov Chain a naive implementation of the algorithm needs more memory than the analogous algorithm using CFTP if used with a seeded pseudo-random number generator. Similarly, our algorithm will also need more memory than the CFTP algorithm in [54]. The CFTP algorithm only needs to store the seed used at each time step;

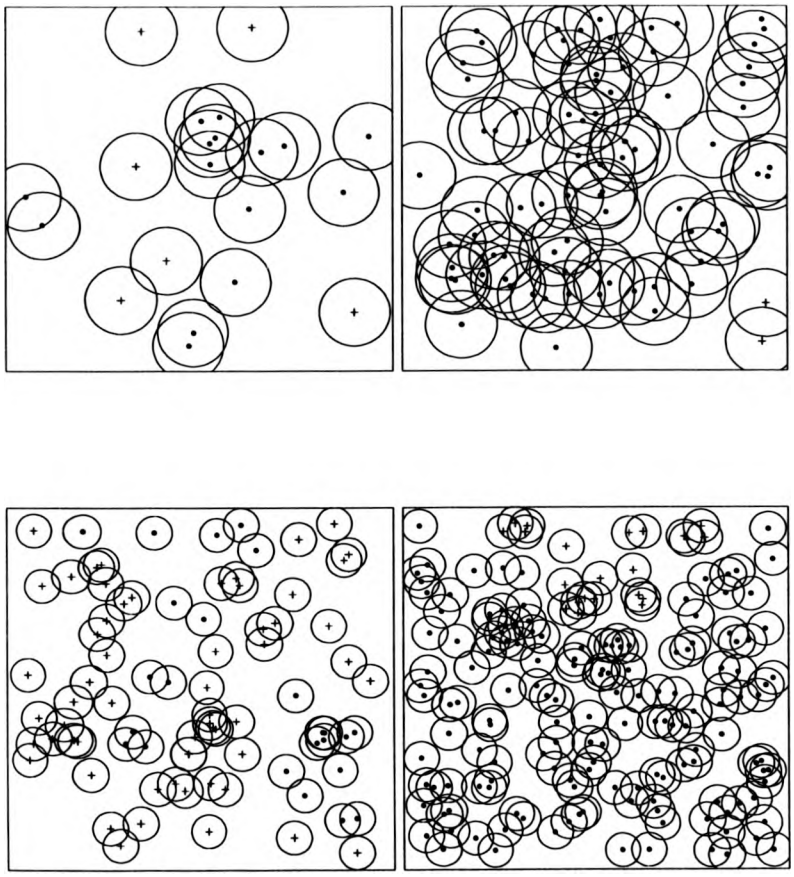

Figure 25: Realisations of the symmetric mixture model with $\beta = 30, R = 0.2$ (top left), $\beta = 80, R = 0.2$ (top right), $\beta = 100, R = 0.1$ (bottom left), $\beta = 200, R = 0.1$ (bottom right). The circles indicate the area in which no point of the other type is allowed. This figure is taken from [140].

it can then reconstruct the Poisson processes which were used at this time step in the previous iteration. For Fill's algorithm however, we need to store the whole time-reversed trajectory, because the seed used at each time-step does not provide enough information to reconstruct the configuration of the time-reversed trajectory at this time-step. See also the code of the programm `Perfectsim` in the Appendix A3.

In the case of a reversible finite state space Markov chain, Fill derives an upper bound on the expected number of Markov transitions, which is smaller than the bound on the expected number of transitions given by Propp and Wilson [111] for CFTP. Both bounds rely strongly on the assumption of having a finite state space. At present there are no bounds on the number of expected transitions for the algorithm in [54]. Thus in order to compare the number of iterations I needed for our algorithm with the number of iterations needed for CFTP we performed some simulation experiments. As with CFTP we started each simulation with one iteration of the Gibbs Sampler and increased the number of Gibbs Sampler iterations by a factor of two if the proposed sample was not accepted. Figure 26 shows the Monte Carlo estimates of the expected number of iterations I of the Rejection Sampling algorithm. To enable the comparison with the Monte Carlo estimates in [54] we plotted the average number of iterations for a symmetric penetrable spheres mixture model versus the parameter $\theta = \log(\beta_1) = \log(\beta_2)$. We

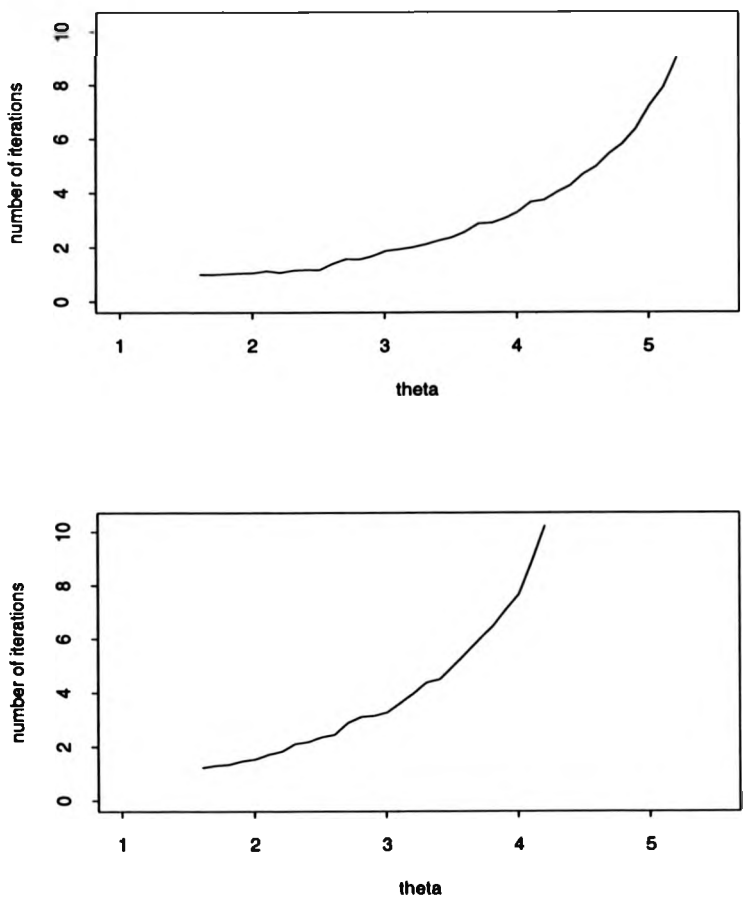

Figure 26: Monte Carlo estimates of the expected number of iterations needed for the symmetric mixture model as a function of θ for interaction radius $R = 0.1$ (top) and $R = 0.2$ (bottom). This figure is taken from [140].

found that the rejection sampling algorithm used about the same number of iterations as CFTP.

7.3.7 The Dual Process

In a similar fashion to Section 7.2 we can define a dual process X^* for the `GibbsSampling1` chain. Let Z_j be a Poisson process on W with intensity $\beta_j, j = 1, 2$, and let $E_2 = \{(x, y) \in W : f(x, y) > 0\}$, where f is the density of the penetrables spheres mixture model. We define $F : E_2 \times \Omega^2 \to E_2$ as

$$F\Big((x, y), Z_1, Z_2\Big) \quad = \quad \Big(Z_1 \backslash [y \oplus B],\ Z_2 \backslash \Big[(Z_1 \backslash [y \oplus B]) \oplus B\Big]\Big),$$

which is a transition rule for `GibbsSampling2`, the time-reversal of `GibbsSampling1`.

Suppose we start in a minimal state $z_0 = (z_1, z_2) = (\emptyset, z_2)$ and perform N iterations of `GibbsSampling1` and thus produce the trajectory

$$(X_0, \ldots, X_N) \quad = \quad (x_0 = z_0, \ldots, x_N)$$

Now let $\hat{Z}_{1,k}$ have the distribution of Z_1 conditional on $F((x_k, y_k), Z_1, Z_2) = (x_{k-1}, y_{k-1})$. Then the process $\hat{Z}_{1,k}$ is the superposition of x_{k-1} and a Poisson process of intensity β_1 on $(y_k \oplus B) \cap W$. Similarly, if $\hat{Z}_{2,k}$ has the distribution of Z_2 conditional on $F((x_k, y_k), Z_1, Z_2) = (x_{k-1}, y_{k-1})$, then the process $\hat{Z}_{2,k}$ is the superposition of y_{k-1} and a Poisson process of intensity β_2 on $(x_{k-1} \oplus$

$B) \cap W$. The dual process X^* is then defined as follows:

$$X_0^* = \{z_0\} \qquad \text{and}$$

$$X_n^* = \left\{(x,y) \in E_2 : F((x,y), \hat{Z}_{1,k}, \hat{Z}_{2,k}) \in X_{n-1}^*\right\} \quad \text{for } n \in \{1,\ldots,N\}.$$

For example, $\hat{Z}_{1,1}$ is a Poisson process Z of intensity β_1 on $(y_1 \oplus B) \cap W$ and $\hat{Z}_{2,1} = z_2$. Thus

$$\begin{aligned} X_1^* &= \left\{(x,y) \in E_2 : Z\backslash[y \oplus B] = z_1,\ z_2\backslash\Big[(Z\backslash[y \oplus B]) \oplus B\Big] = z_2\right\} \\ &= \left\{(x,y) \in E_2 : Z \subseteq (y \oplus B) \cap W\right\}, \end{aligned}$$

and so $X_1^* = E_2$ if $Z = \emptyset$ which occurs with probability $\exp(-\beta_1 \lambda_L(y_1 \oplus B))$.

Consider an alternative method of producing an exact sample of the penetrable spheres mixture model. We could first produce a sample y of a Poisson process of intensity β_2 on W. Then we produce a sample x of a Poisson process of intensity β_1 on W. We accept (x,y) as a sample from the mixture model if $x \cap (y \oplus B) = \emptyset$. This happens with probability $\exp(-\beta_1 \lambda_L([y \oplus B])$. We will refer to this procedure as rejection sampling in the usual sense.

Now reconsider our algorithm. If we fix the run time of each iteration to $N = 1$, we would perform the following procedure. We start `GibbsSampling1` from z_0 and produce the sample (x_1, y_1). Then y_1 is a realisation of a Poisson process of intensity β_2 on W and x_1 is produced by sampling a Poisson process of intensity β_1 and thinning out any point in $y_1 \oplus B$. This sample is then accepted if $X_1^* = E_2$, that is with probability $\exp(-\beta_1 \lambda_L(y_1 \oplus B))$. But this

is exactly the probability that a Poisson process of intensity β_1 does not put any points in $y_1 \oplus B$. Thus for $N = 1$ our rejection sampling algorithm is just rejection sampling in the usual sense. However, by increasing the run time N we increase the probability of accepting and thus get a higher acceptance probability than for rejection sampling in the usual sense.

One might consider using the dual process X^* directly rather than Fill's algorithm. However, the expressions for X_n^* become rather complex for $n > 1$, thus using the dual process directly to check whether X has reached stationarity is not practical.

7.3.8 Conclusions

This chapter shows that, as with CFTP, Fill's rejection sampling algorithm is available and practical for some point processes. We have illustrated how Fill's algorithm can be combined with the Gibbs sampler in [54] to produce exact samples of the penetrable spheres mixture process and associates. The output produced by the algorithm is independent of the run time of the algorithm. Note, however that we measure the run time in terms of Markov transitions and not computer time. For the penetrable spheres mixture model the number of points is unbounded and thus each Markov transition may need an arbitrary amount of computer time. Thus our algorithm is interruptible with respect to the number of Markov chain steps but not with respect to

time.

Note that this application of Fill's algorithm provides a *new* perfect simulation algorithm for point processes. Simulation suggests that this new algorithm is approximately as efficient as the analogous algorithm using CFTP. Our algorithm needs more memory than CFTP, but protects against bias caused by user impatience. This is a significant virtue: however the main contribution of this section is to show how one can perform perfect simulation for various point processes in quite a different manner from CFTP.

The Gibbs sampler used is restricted to attractive versions of the penetrable spheres mixture process and related models. It would be useful to develop Fill's version of rejection sampling in cases where dominated CFTP can be applied. In particular it would be useful to have a rejection sampling algorithm which can produce samples of the attractive area-interaction. One major difficulty is that in these case the maximal and minimal process are stochastically varying. At present we cannot see how to overcome this barrier.

7.4 Further Extensions of Fill's Algorithm

In this section we present some extensions of Fill's algorithm to cases where we do not have a monotone transition matrix $\tilde{P}$.

7.4.1 Fill's Algorithm in the General Setting

Fill's algorithm samples a Markov chain X when its set-valued dual X^* is in its absorbing state, which is the order ideal which is identical to the state space E. Let

$$(X_0, \ldots, X_N) \quad = \quad (x_0, \ldots, x_N)$$

be a trajectory of X, Suppose $\bar{f}(\cdot, U)$ is a transition rule for $\bar{P}$ and the distribution of $\hat{U}_n$ is the conditional distribution of U given that $\bar{f}(x_n, U) = x_{n-1}$. We start X^* in the initial state $\{x_0\}$ and set

$$X_n^* \quad = \quad \Big\{ y \in E : \bar{f}(\cdots(\bar{f}(\bar{f}(y, \hat{U}_n), \hat{U}_{n-1})\cdots), \hat{U}_1) = x_0 \Big\}.$$

for $n \geq 1$. Then X^* is a dual process for X. Now, in the monotone case we set $x_0 = \hat{0}$ and deduce that $X_N^* = E$ if

$$\bar{Y}_N \quad = \quad \bar{f}(\cdots \bar{f}(\bar{f}(\hat{1}, \hat{U}_N), \hat{U}_{N-1})\cdots), \hat{U}_1) \quad = \quad \hat{0}.$$

But what if we do not have a monotone transition rule? This case was considered by Fill (unpublished) and in a slightly different setting by Murdoch and Rosenthal [100]. If we do not have monotonicity, we have to check for each state $y \in E$ that

$$\bar{f}(\cdots \bar{f}(\bar{f}(y, \hat{U}_N), \hat{U}_{N-1})\cdots), \hat{U}_1) \quad = \quad x_0.$$

The rejection sampling algorithm would then work as follows. Suppose the algorithm `Markov(x)` draws from $P(x, \cdot)$ and that `CondRandom(x,y)` draws

from the conditional distribution of U given $\tilde{f}(x, U) = y$.

```
GeneralFill(N):
   x_0 ← x_0
   for k = 1 to N
      x_k ← Markov (x_{k-1})
   for k = 0 to N
      x̃_k ← x_{N-k}
   ỹ_0 ← E
   for k = 1 to N
      u ← CondRandom(x̃_{k-1}, x̃_k)
      ỹ_k ← image{f̃(ỹ_{k-1}, u)}
   if (ỹ_N = {x_0})
      return(x_N)
   else
      GeneralFill(2N)
```

The proof of the correctness of the algorithm is similar to the proof in the monotone setting. Let

$$F_0^N = \text{image}\{\tilde{f}(\cdots \tilde{f}(\tilde{f}(\cdot, \tilde{U}_N), \tilde{U}_{N-1}) \cdots), \tilde{U}_1)\}.$$

As in the monotone setting, Fill's algorithm draws a state z from $P^N(x_0, \cdot)$ and proposes it as a sample of $\pi(\cdot)$. The sample z must then be accepted with probability

$$\frac{1}{c} \frac{\pi(z)}{P^N(x_0, z)},$$

where c is an upper bound on the ratio

$$\frac{\pi(z)}{P^N(x_0, z)}.$$

According to the definition of $\tilde{P}$ and we have that

$$\frac{\pi(z)}{P^N(x_0,z)} \quad = \quad \frac{\pi(x_0)}{\tilde{P}^N(z,x_0)} \quad \leq \quad \frac{\pi(x_0)}{\mathbb{P}(F_0^N = x_0)}.$$

Thus

$$c \quad = \quad \frac{\pi(x_0)}{\mathbb{P}(F_0^N = x_0)}$$

is a valid choice for a rejection sampling procedure. The sample z then has to be accepted with probability

$$\frac{1}{c}\,\frac{\pi(z)}{P^N(x_0,z)} \quad = \quad \frac{\mathbb{P}(F_0^N = x_0)}{\pi(x_0)}\,\frac{\pi(x_0)}{\tilde{P}^N(z,x_0)} \quad = \quad \frac{\mathbb{P}(F_0^N = x_0)}{\tilde{P}^N(z,x_0)}.$$

For now, assume we have a $\tilde{P}$-trajectory $(\tilde{X}_0 = \tilde{x}_0, \ldots, \tilde{X}_n = \tilde{x}_n)$ and that the distribution of $\hat{U}_n$ is the conditional distribution of U given $\tilde{f}(\tilde{x}_{n-1}, U) = x_n$. Then

$$\mathbb{P}(F_0^N = x_0) \;=\; \prod_{x \in E} \Big(\mathbb{P}(\tilde{f}(\cdots, \tilde{f}(\tilde{f}(x, \hat{U}_N), \hat{U}_{N-1})\cdot, \hat{U}_1) = x_0)\Big).$$

If we produce the trajectory $(\tilde{X}_0 = z, \ldots, \tilde{X}_N = \hat{0})$ by reversing a P-trajectory in time, then it has the distribution of a $\tilde{P}$-trajectory which is conditioned to start in z and end in x_0. Thus the algorithm accepts the sample with probability

$$\frac{\mathbb{P}(F_0^N = x_0)}{\tilde{P}^N(z,x_0)}.$$

However, although Fill's algorithm can be defined for chains with non-monotone transition kernels, it is rather inefficient and in particular is not feasible for large state spaces.

7.4.2 Fill's Algorithm in the Anti-Monotone Setting

Møller and Schladitz [96] extended Fill's algorithm to deal with the case, when $\tilde{P}$ has a transition rule which is anti-monotone rather than monotone. They define a maximal chain $\tilde{Y}^{\max}$ and a minimal chain $\tilde{Y}^{\min}$ such that if

$$\tilde{Y}_N^{\max} \quad = \quad \tilde{Y}_N^{\min} \quad = \quad \hat{0}$$

then for all $y \in E$ we have

$$\tilde{f}(\cdots \tilde{f}(\tilde{f}(y, \hat{U}_N), \hat{U}_{N-1}) \cdots), \hat{U}_1) \quad = \quad \hat{0}.$$

In [96] the algorithm is developed for continuous state spaces with quasiminimal and quasimaximal elements. We present the algorithm in the simpler case of a finite state space and describe it using pseudocode. Suppose the algorithm `Markov(x)` draws from $P(x, \cdot)$ and that `CondRandom(x,y)` draws from the conditional distribution of U given $\tilde{f}(x, U) = y$, where $\tilde{f}$ is an anti-monotone transition rule for $\tilde{P}$.

```
AntiMonotoneFill(N):
   x_0 ← 0̂
   for k = 1 to N
      x_k ← Markov (x_{k-1})
   for k = 0 to N
      x̃_k ← x_{N-k}
   ỹ_1^min ← 0̂
   u ← CondRandom(x̃_0, x̃_1)
   ỹ_1^max ← f̃(0̂, u)
   for k = 2 to N
      u ← CondRandom(x̃_{k-1}, x̃_k)
```

```
    ȳ_k^min ← f̃(ȳ_{k-1}^max, u)
    ȳ_k^max ← f̃(ȳ_{k-1}^min, u)
if (ȳ_N^max = 0̂)
    return(x_N)
else
    AntiMonotoneFill(2N)
```

As $\tilde{f}$ is anti-monotone, for any $y \in E$

$$\tilde{f}(\hat{0}, \hat{U}_1) \quad = \quad \tilde{Y}_1^{\max} \quad \geq \quad \tilde{f}(y, \hat{U}_1)$$

A cross-over then ensures that for $n \geq 2$

$$\tilde{Y}_n^{\min} \quad \leq \quad \tilde{f}(\cdots \tilde{f}(\tilde{f}(y, \hat{U}_n), \hat{U}_{n-1}) \cdots), \hat{U}_1) \quad \leq \quad \tilde{Y}_n^{\max} \qquad \text{for all } y \in E.$$

Thus the maximal chain and the minimal chain can be used to monitor when the dual process X^* has reached equilibrium.

8 Conclusions and Further work

This thesis discussed applications and new developments of simulation methods in stochastic geometry. Chapter 3 illustrated the use of simulation methods for the analysis of spatial point patterns. Conclusions from the simulation study were drawn in Section 3.6.

In Chapters 5, 6 and 7 we presented a new variant of Markov chain Monte Carlo methods: perfect simulation algorithms. These algorithms are a recent development, thus many open question still remain.

Two type of perfect simulation methods are available to date: Coupling from the Past and Fill's interruptible algorithm. In Chapter 5 we presented with Theorem 5.2 the most general form of Coupling from the Past (CFTP) available to date. Theorem 5.2 assumes an underlying process from which we derive both the target Markov chain and the bounding process. However, in Chapter 6 we defined a minimal and maximal process, and thus a bounding process, and then derived a realisation of an ergodic Markov chain from a realisation of the minimal and maximal process. Thus the target Markov chain was coupled to the bounding process. This shows that we could present a Coupling from the Past set-up which is just based on requirements of the bounding process without refering to an underlying process as follows. Suppose we would like to sample the distribution π on the state space E.

Proposition 8.1 *Suppose there exists a bounding process Y on $\mathcal{P}(E')$, the subsets of some state space $E' \supseteq E$, which satisfies the following requirements: Firstly it needs to possess the funneling property:*

$$Y_{-s}^{-T-u} \subseteq Y_{-s}^{-T} \qquad \text{for all } -u \leq 0 \text{ and } -T \leq -s \leq 0.$$

Secondly, we assume the bounding process has the following coalescence properties:

$$\text{if } Y_{-s}^{-T} \text{ is a singleton, then } Y_{-u}^{-T} \text{ is a singleton for } -s \leq -u \leq 0$$

and

$$T_C = \inf\{T \geq 0 : Y_0^{-T} \text{ is a singleton } \} < \infty \quad \text{almost surely.}$$

Finally, we require that there is an ergodic Markov chain which is coupled to Y as follows:

$$X_{-s}^{-T}\big(g(Y_{-T}^{-T})\big) = G\big(\{Y_{-t}^{-T}, -T \leq -t \leq -s\};\ g(Y_{-T}^{-T})\big)$$

where $\{g(Y_{-t}^{-T}), -t \geq -T\}$ is a time-homogeneous process on E. The process $\{X_{-s}^{-T}(g(U_{-T})), -s \geq -T\}$ is assumed to behave like an ergodic Markov chain with equilibrium distribution π started at time $-T$ in the state $g(Y_{-T}^{-T})$. Set

$$T_C = \inf\{\, T : Y_0^{-T} \text{ is a singleton } \},$$

then $Y_0^{-T_C}$ has the equilibrium distribution π.

We decided not to present Theorem 5.2 in this more general form as the derivation of a CFTP algorithm is often more transparent when there is explicit dependence on an underlying process. However, when developing the CFTP algorithm in [74], Kendall observed that the above form enables us to define more general CFTP algorithms. The above Proposition allows the bounding process Y^{-T-S}, where $-S \leq 0$, to use *additional* random variables or processes on $[-T, 0]$ which are independent of the random variables or processes used by Y^{-T}. This type of construction may enable us to produce Boolean models conditioned to satisfy a connectivity constraint; we plan to investigate this further in joint work with Kendall.

The alternative method of perfect simulation, Fill's interruptible algorithm, has received less attention than CFTP. Fill [37, 38] compares the space and time requirements of CFTP with his interruptible algorithm, see also Section 7.3.6. It would be useful to have a more detailed comparison of CFTP with the interruptible algorithm.

As pointed out in the previous chapter, the interruptible algorithm has the advantage over CFTP that it is protected against user impatience bias. We also showed that for some Markov chains user impatience bias cannot occur, see Section 5.5.4. It would be of interest to identify transition kernels for which the protection against user impatience is automatic.

CFTP requires realizable monotonicity whereas the interruptible algo-

rithm only needs stochastic monotonicity, see the remarks in Section 4.2. Fill and Machida [39] discuss conditions when stochastic monotonicity is equivalent to realizable monotonicity. The authors identified examples for transition kernels which are stochastically monotone but not realizable monotone, however, none of these examples are of any practical relevance. So far, there is no Fill-type algorithm which could not be translated into an analogous CFTP algorithm. Further investigation is called for.

The applicability of Fill's algorithm would be significantly increased if it could be combined with the methods which are used for dominated CFTP. But, as pointed out in Chapter 7, we have not yet been able to combine the interruptibe algorithm with these methods.

Now that perfect simulation algorithms are available, methods need to be developed which make efficient use of the samples generated. Murdoch and Rosenthal [99] developed first steps in this direction.

The underlying theory of the present perfect simulation methods, CFTP and Fill's algorithm, are not new but were used for some considerable time to assess convergence of Markov chains. One major open question is whether there are any further methods which could be used to develop perfect simulation algorithms. However, until there is an answer to this question, it seems sensible to further explore and expand the current methods.

A Simulation Code

A.1 Simulations in Chapter 3

Following is part of the S-code for the simulation study in Chapter 3. The first module produces a Poisson point pattern.

```
poisson.pts <- function(m, k)
{
# Samples Poisson point pattern with intensity = m and seed = k
        set.seed(k)
        n <- rpois(1, m)
        x <- runif(n)
        y <- runif(n)
        pts <- matrix(NA, length(x), 2)
        pts[, 1] <- x
        pts[, 2] <- y
        pts
}
```

The following module estimates the F-, G- and J-function of the sampled Poisson patterns for the interval points in `breaks`. It uses the submodules `min.distanz.pts` which computes the minimum distances between incidents, `min.distanz.obj` which computes the minimum distances from a sampling point to an incident and `estimation` which computes the empirical distribution functions of the minimum distances.

```
statistiken.pois <- function(m, l, k)
{
# Simulates m Poisson patterns of intensity l with inital
# seed k, then estimates the $G$-, $F$- and $G$-function
        G.f <- matrix(NA, m, length(breaks))
        F.f <- matrix(NA, m, length(breaks))
        J.f <- matrix(NA, m, length(breaks))
        for(n in 1:m) {
                pts <- poisson.pts(l, n + k)
                punkte.dist <- min.distanz.pts(pts)
                objekte.dist <- min.distanz.obj(pts)
                G.est <- rep(100, length(breaks))
```

```
                F.est <- rep(100, length(breaks))
                J.est <- rep(100, length(breaks))
                G.f[n,  ] <- estimation(breaks, punkte.dist)
                F.f[n,  ] <- estimation(breaks, objekte.dist)
                J.f[n,  ] <- (1 - G.f[n,  ])/(1 - F.f[n,  ])
        }
        return(F.f, G.f, J.f)
}
```

The submodule `estimation`:

```
estimation <- function(breaks, distanzen)
{
# Given the distances computes the empirical
# distribution function
        dist <- distanzen
        est <- rep(NA, length(breaks))
        for(i in 1:length(breaks)) {
        if(max(dist) > breaks[i]) {
        dist <- dist[dist > breaks[i]]
        est[i] <- (length(distanzen)-length(dist))
                  /length(distanzen)
                }
                else est[i] <- 1
        }
        est
}
```

This module produces samples of a Matérn cluster process:

```
matern <-function(l, mu, R, k)
{
# Produces Matern cluster pattern with l = intensity,
# mu = mean number of children,  R = interaction radius,
# k = seed
        set.seed(k)
        N <- (1 + 2 * R)^2 * l
        s <- trunc(N/mu) + 1
        s <- rpois(1, s)
```

```
        v <- (sqrt(N/l) - 1)/2
        x <- (1 + 2 * v) * runif(s) - v
        y <- (1 + 2 * v) * runif(s) - v
        n <- rpois(s, mu)
        t <- sum(n)
        r <- runif(t)
        r <- sqrt(r * (R^2))
        w <- 2 * pi * runif(t)
        children.pts <- matrix(NA, t, 2)
        children.pts[, 1] <- r * cos(w)
        children.pts[, 2] <- r * sin(w)
        h <- 1
        for(i in 1:s) {
            if(n[i] > 0) {
              for(j in h:(h - 1 + n[i])) {
                children.pts[j, 1] <- children.pts[j, 1] + x[i]
                children.pts[j, 2] <- children.pts[j, 2] + y[i]
                        }
                }
                h <- h + n[i]
        }
        ind <- (children.pts >= 0) & (children.pts <= 1)
        ind <- ind + 0
        p <- pmin(ind[, 1], ind[, 2])
        pts <- matrix(NA, length(p[p > 0]), 2)
        pts[, 1] <- children.pts[, 1][p > 0]
        pts[, 2] <- children.pts[, 2][p > 0]
        pts
}
```

By exchanging `matern` for `poisson.pts` we can use `statistiken.pois` to produce empirical F-, G- and J-functions of Matérn cluster patterns. The following module produces Thomas cluster samples.

```
thomas.cluster <- function(mu, sigma, s)
{
# Produces Thomas field pattern with
# mu = mean number of children,
# sigma = location standard deviation, s = seed
```

```
   set.seed(s)
   rho <- floor( 100/(mu + 1) )
   parents <- list( x=runif(rho), y=runif(rho) )
   parent.ind <- floor(runif(100-rho, min=1, max = rho + 1))
   pos <- list( x=rnorm(100-rho, mean=0, sd=sigma),
y=rnorm(100-rho, mean=0, sd=sigma) )
   ptt <- parents
   ptt$x <- c(ptt$x, pos$x + parents$x[parent.ind])
   ptt$y <- c(ptt$y, pos$y + parents$y[parent.ind])
# map to torus
  for(m in 1:100) {
   if( ptt$x[m]<0) ptt$x[m]<-ptt$x[m]-floor(ptt$x[m])
   if( ptt$x[m]>1) ptt$x[m]<-ptt$x[m]-floor(ptt$x[m])
   if( ptt$y[m]<0) ptt$y[m]<-ptt$y[m]-floor(ptt$y[m])
   if( ptt$y[m]>1) ptt$y[m]<-ptt$y[m]-floor(ptt$y[m])
   }
  return(ptt)
}
```

The module `hard.core` produces hard core patterns. The submodule `thin` performs the neccessary thinning.

```
hard.core <- function(lambda, h, k)
{
# Produces a hard core pattern of intensity lambda,
# h = hardcore distance,  k = seed

# SIMULATION OF UNDERLYING BINOMIAL PROCESS
        if(1 - pi * lambda * h^2 > 0) {
          set.seed(k)
                lb <-  - log(1 - pi * lambda * h^2)/(pi * h^2)
                N <- (1 + 2 * h)^2 * lb
                s <- trunc(N) + 1
                s <- rpois(1, s)
                v <- (sqrt(s/lb) - 1)/2
                x <- (1 + 2 * v) * runif(s) - v
                y <- (1 + 2 * v) * runif(s) - v
# THINNING PROCEDURE
                p <- thin(x, y, h)
```

```
# DETERMINING POINTS WHICH ARE WITHIN THE SAMPLE WINDOW
                ind <- (x>=0)&(x<=1)&(y>=0)&(y<=1)
                ind <- ind + 0
                r <- pmin(ind, p)
                pts <- matrix(NA, length(r[r > 0]), 2)
                pts[, 1] <- x[r > 0]
                pts[, 2] <- y[r > 0]
                pts
        }
        else print("not defined")
}
```

We used the modules maxi and integ to calculate the Maximum and Integral Statistic.

```
maxi <- function(data)
{
# Determines a maximum in a range of values
     maxwert <- matrix(0,length(data[,1]),length(data[1,]))
     maxwert[, 1] <- data[, 1]
     for(i in 2:(length(data[1,  ]))) {
            maxwert[, i] <- pmax(data[, i], maxwert[, i - 1])
     }
     maxwert
}

integ <- function(fkt)
{# Computes integral of the sqare of the given function
   integral <- matrix(0, length(fkt[, 1]), length(fkt[1,  ]))
   integrant <- matrix(NA, length(fkt[, 1]), length(fkt[1,  ]))
     for(i in 1:length(fkt[, 1])) {
       integrant[i,  ] <- fkt[i,  ]^2
     }
     integral[, 1] <- rep(0, length(fkt[, 1]))
      for(i in 2:length(fkt[1,  ])) {
       integral[, i] <- integral[,i - 1] + integrant[,i - 1]
     }
   integral
}
```

A.2 Perfect Simulation of a Conditional Boolean model

This is the S-code which as used used to produce the samples of the conditional Boolean model in Chapter 6.

```
boolean <- function(S, s, cond.pts, mu, W, rad)
{
# S = starting time, s = seed, mu = intensity,
# rad = grain radius, cond.pts = conditioning points,
# W = windowsize,
    K <- length(pts)/2
    cyl <- make.cylinder(pts, mu, S, s, W, rad)
    event <- make.events(cyl, K, S, s)
    min.max <- configs(cyl, K, S)
    cat("transitions in total:", length(event[, 1]), fill = T)
    for(n in 1:(length(event[, 1]))) {
        if(event[n, "type"] == 1)
        {
          min.max <- birth(cyl[event[n,"index"],], min.max, K)
        }
        if(event[n, "type"] == 2)
        {
           if(cyl[event[n, "index"], "type"] == 3)
           {
             min.max<-death(3,event[n,"index"],min.max, K, rad)
   }
           if(cyl[event[n, "index"], "type"] == 1)
           {
             min.max<-death(1,event[n,"index"],min.max, K, rad)
           }
        }
        if(event[n, "type"] == 3)
        {
          min.max <- death(2,event[n,"index"],min.max, K, rad)
        }
    }
    return(min.max, cyl)
}
```

```
make.cylinder <- function(cond.pts, mu, S, s, W, rad)
{
# Creates list of cylinders, whose birthtime is before 0
# and whose deathtime is after S (S is negative)
# mu = birthrate, S= starting time, s=seed, W=windowsize
    set.seed(s)
    K <- length(cond.pts)/2
    cyl <- NULL
    cyl.vec <- rep(NA, (8 + K))
    names(cyl.vec) <- c("death", "life", "xloc", "yloc",
        "radius","type", "index", rep("cover", K), "nummer")
    pois.pt <- rexp(1, mu) - 1
    N <- 0
    while(pois.pt < 0) {
        cyl.vec["death"] <-  - log(1 + pois.pt)
        cyl.vec["life"] <- cyl.vec["death"] + rexp(1, 1)
        cyl.vec["xloc"] <- runif(1, 0, W)
        cyl.vec["yloc"] <- runif(1, 0, W)
        cyl.vec["radius"] <- rad
        cyl.vec["type"] <- 1
        cyl.vec["index"] <- N + 1
        cyl <- rbind(cyl, cyl.vec)
        N <- N + 1
        pois.pt <- pois.pt + rexp(1, mu)
        }
    while(pois.pt + S < 0) {
        cyl.vec["death"] <-  - pois.pt
        cyl.vec["life"] <- rexp(1, 1)
        cyl.vec["xloc"] <- runif(1, 0, W)
        cyl.vec["yloc"] <- runif(1, 0, W)
        cyl.vec["radius"] <- rad
        cyl.vec["type"] <- 1
        cyl.vec["index"] <- N + 1
        cyl <- rbind(cyl, cyl.vec)
        N <- N + 1
        pois.pt <- pois.pt + rexp(1, mu)
        }
        cyl <- rbind(cyl, virtual(cond.pts, 2 * rad, S))
```

```
        cyl[, 8:(7 + K)] <- list.cover(cyl, cond.pts)
        for(j in 1:length(cyl[, 1]))
        {
            if(max(cyl[j, 8:(7 + K)]) == 0) {
                cyl[j, "type"] <- 3}
            cyl[, "nummer"] <- seq(1:(N + K))
        }
    cyl
}

virtual <- function(cond.pts, rad, S)
{
# Creates virtual grains
    K <- length(cond.pts)/2
    cyl <- matrix(NA, length(cond.pts[, "xloc"]), (8 + K))
    dimnames(cyl)<-list(rep("virtual",
          length(cond.pts[,"xloc"])),c("death", "life",
            "xloc", "yloc", "radius",
            "type", "index", rep("cover", K), "nummer"))
    cyl[, 1] <- rep(0, length(cond.pts[, "xloc"]))
    cyl[, 2] <- rep(abs(S) + 1, length(cond.pts[, "xloc"]))
    cyl[, 3:4] <- cond.pts
    cyl[, 5] <- rep(rad, length(cond.pts[, "xloc"]))
    cyl[, 6] <- rep(2, length(cond.pts[, "xloc"]))
    cyl[, 7] <- seq(1:length(cond.pts[, "xloc"]))
    cyl
}

make.events <- function(cyl, K, S, s)
{
# produces a list of birth and death times including death
# times for perpetuated grains,
# S = starting time, cyl = matrix of cylinders,
# K = number of conditioning points, s = seed
    set.seed(s + 1)
    N <- length(cyl[, 1]) - K
    events <- matrix(NA, 2 * N, 3)
    dimnames(events) <- list(NULL, c("time", "type", "index"))
```

```
    events[1:N,  ] <- c(cyl[1:N,"death"] - cyl[1:N,"life"],
                                    rep(1, N), 1:N)
    events[(N+1):(2*N),] <- c(cyl[1:N,"death"],rep(2,N),1:N)
    perpet.death <- 0
    while(perpet.death > S) {
        life <- rexp(1, K)
        perpet.death <- perpet.death - life
        index <- as.integer(K * runif(1)) + 1
        events <- rbind(events, c(perpet.death, 3, index))
        }
    events <- events[sort.list(events[, 1]),  ]
    events <- events[(events[,"time"]>S)&(events[,"time"]<0),]
    events
}

configs <- function(cyl, K, S)
{
# Inital configurations
    MAX <- config.max(cyl, K, S)
    MIN <- config.min(MAX, K)
    index.min <- index(MIN[, 4:(3 + K)])
    index.max <- index(MAX[, 4:(3 + K)])
    return(MAX, MIN, index.min, index.max)
}

config.max <- function(cyl, K, S )
{
# Initial maximum configuration
    p <- (cyl[,"death"] - cyl[,"life"] < S) & (cyl[,"death"]>S)
    MAX <- cyl[p, 5:(8 + K)]
    MAX
}

config.min <- function(MAX, K)
{
# Initial minimum configuration
    MIN <- MAX[1:(length(MAX[, "type"]) - K),  ]
    MIN
}
```

```
index <- function(cover.mat)
{
# Marks covering grains
    index <- rep(NA, length(cover.mat[1,  ]))
    for(i in 1:length(cover.mat[1,  ]))
        index[i] <- sum(cover.mat[, i])
    index
}

list.cover <- function(cyl, cond.pts)
{
# outputs for each cylinder the conditional pts it covers
    N <- length(cyl[, "xloc"])
    K <- length(cond.pts[, "xloc"])
    cover <- matrix(NA, N, K)
    for(j in (1:N)) {
        cover[j,  ] <- ((cyl[j, "xloc"] - cond.pts[, "xloc"])^2
             + (cyl[j, "yloc"] -
             cond.pts[, "yloc"])^2 < cyl[j, "radius"])^2)
    }
    cover
}

birth <- function(grain, config, K)
{
    config$MAX <- rbind(grain[5:(8 + K)], config$MAX)
    config$MIN <- rbind(grain[5:(8 + K)], config$MIN)
    config$index.min <- config$index.min + grain[8:(7 + K)]
    config$index.max <- config$index.max + grain[8:(7 + K)]
    config
}

death  <- function(grain.type, grain.index, config, K, rad)
{
    D.min <- 100
    D.max <- 100
# check whether and where grain in MIN and MAX
    grain.index.max <- row(config$MAX)[(config$MAX[, "type"] ==
```

```
            grain.type) &
            (config$MAX[,"index"] == grain.index),1]
    grain.index.min <- row(config$MIN)[(config$MIN[, "type"] ==
            grain.type) &
            (config$MIN[,"index"] == grain.index),1]
    if(!(length(grain.index.max) == 0)) {
        cover.max <- config$MAX[grain.index.max, 4:(3 + K)]
# death if grain does not cover any cond. pt
        if(config$MAX[grain.index.max, "type"] == 3) {
            D.min <- 1
            D.max <- 1
        }
# death if grain covers at least one cond. pt
        if(!(config$MAX[grain.index.max, "type"] == 3)) {
# case 1: grain in MAX, but not in MIN
        if(length(grain.index.min) == 0) {
            D.min <- 2
            ind <- config$index.min[cover.max > 0]
            if(min(ind) > 0) {  D.max <- 1 }
            if(min(ind) == 0) { D.max <- 0 }
        }
# case 2: grain in MAX and in MIN
        if(!(length(grain.index.min) == 0)) {
# check MIN
            cover.min <- config$MIN[grain.index.min, 4:(3 + K)]
            ind <- config$index.max[cover.min > 0]
            cover.min[cover.min > 0]
            if(min(ind) > 0) { D.min <- 1 }
            if(min(ind) == 0) { D.min <- 0 }
# check MAX
            ind1 <- config$index.min[((cover.max > 0) &
                    (cover.min > 0))]-cover.max[((cover.max>0)
                    & (cover.min > 0))]
            ind2 <- config$index.min[((cover.max > 0) &
                    (cover.min == 0))]
            if(length(ind2) == 0) {
                if(min(ind1) == 0) { D.max <- 0 }
                if(min(ind1) > 0) { D.max <- 1 }
```

```
        j <- 1
        flag <- 1
        while((flag > 0) & (j <= length(Ps.vec))) {
            flag <- sum(pois.proc == Ps.vec[j])
            j <- j + 1
        }
        if(flag == 0) {
           config$MAX[grain.index.max, "type"] <- 2
           config$MAX[grain.index.max,"index"] <- Ps.vec[j-1]
           if(D.min == 0) {
            config$MIN[grain.index.min,"index"]<-Ps.vec[j-1]
            config$MIN[grain.index.min, "type"]<-2
           }
        }
# no poisson process free, we join grains
        if(flag > 0) {
# which Ps process is free in min ?
          pois.proc.min -config$MIN[(config$MIN[,"type"]==2),
                                          "index"]
          del <- NULL
          if(length(pois.proc.min) > 0) {
               for(i in 1:length(Ps.vec)) {
                   ind <- (Ps.vec[i] == pois.proc.min)
                   if(sum(ind) > 0)
                   del <- c(del, i)
               }
          }
             if(!is.null(del)){
                Ps.vec <- Ps.vec[ - del]
                join.grain <- Ps.vec
                for(i in 1:(length(Ps.vec))) {
                    join.grain[i] <- row(config$MAX)
                       [((config$MAX[, "type"] == 2) &
                       (config$MAX[,"index"] == Ps.vec[i])), 1]
                }
                for(l in 1:length(join.grain)) {
                    config$MAX[join.grain[l], 4:(3 + K)] <-
                        pmin(rep(1, K),
```

```
                        config$MAX[join.grain[l],4:(3 + K)]
                        +config$MAX[grain.index.max,4:(3 + K)])
                    config$MAX[join.grain[l],"radius"]<-2*rad
                }
                config$MAX <- config$MAX[ - grain.index.max,]
                config$index.max<-index(config$MAX[,4:(3 + K)])

            }
        }
config
}
```

A.3 Perfect Simulation of the Penetrable Spheres Mixture Model

This is the C-code `Perfectsim` which produced the penetrable spheres mixture model samples in Chapter 7.

```
#include <stdlib.h>
#include <stdio.h>
#include <math.h>

#define Dist^2(a,b,c,d) (((a-c)*(a-c))+((b-d)*(b-d)))

/* Initialising the variables and functions

double exp(double x);
double drand48();
int Poisson(b)
double b;
{
double u;
double grenze = b;
double wert = 1.0;
int n = 0;
while( wert > grenze ) {
       u = drand48();
       wert *= u;
```

```
       n++;
    }
    n--;
  return n;
}

/* Function which simulates the rejection sampling step

void Backward(bpts1x, bpts1y, bnpts1,
      bpts2x, bpts2y, bnpts2,
      fpts1x, fpts1y, fnpts1,
         fpts2x, fpts2y, fnpts2,
      rpts1x, rpts1y, rnpts1,
       S)

   double bpts1x[], bpts1y[];
   int bnpts1;
   double bpts2x[], bpts2y[];
   int bnpts2;
   double fpts1x[], fpts1y[];
   int fnpts1;
   double fpts2x[], fpts2y[];
   int fnpts2;
   double rpts1x[], rpts1y[];
   int rnpts1;
   double S;
   {
int i, j,  m;
  bnpts1 = 0;
 if( fnpts1 > 0 )  for(m=0; m<fnpts1; m++) {
    j=0;
    while(
     (Dist^2(fpts1x[m],fpts1y[m],bpts2x[j],bpts2y[j])>S*S)
       && (j < bnpts2 )) {
          j++;}
    if( j == bnpts2 ) {
      bpts1x[bnpts1] = fpts1x[m];
      bpts1y[bnpts1] = fpts1y[m];
```

```
      (bnpts1)++;
    }
 }
if( rnpts1 > 0 )  for(m=0; m<rnpts1; m++) {
    j=0;
    while(
     (Dist^2(rpts1x[m],rpts1y[m],bpts2x[j],bpts2y[j])>S*S)
       && (j < bnpts2 )) {
          j++;}
    if( j == bnpts2 ) {
      bpts1x[bnpts1] = rpts1x[m];
      bpts1y[bnpts1] = rpts1y[m];
      (bnpts1)++;
    }
 }
 bnpts2 = 0;
 if( fnpts2 > 0 ) for(m=0; m<fnpts2; m++) {
    j=0;
    while(
      (Dist^2(fpts2x[m],fpts2y[m],bpts1x[j],bpts1y[j])>S*S)
       && (j < bnpts1 )) j++;
    if( j == bnpts1 ) {
      bpts2x[bnpts2] = fpts2x[m];
      bpts2y[bnpts2] = fpts2y[m];
      (bnpts2)++;
    }
 }
}

/* Function which performs the Rejection Sampling algorithm

double RejSamp(intensity,S,seed,T)
   double intensity, S;
   int seed;
   int T;
{
/* Initialising of variables
```

```
double beta = exp(-intensity);
int s = seed;
int i,j,k, n1, m;
        int gridsize = (int)( (1/S + 1) * (1/S + 1) );
int maxnumber = gridsize + (int)(intensity);
double bpts1x[maxnumber], bpts1y[maxnumber];
    int bnpts1;
    double bpts2x[maxnumber], bpts2y[maxnumber];
        int bnpts2;
int no;
double folge1x[T+1][maxnumber], folge1y[T+1][maxnumber];
int folgen1[T+1];
double folge2x[T+1][maxnumber], folge2y[T+1][maxnumber];
int folgen2[T+1];
double folgerx[T][maxnumber], folgery[T][maxnumber];
int folgenr[T];
double a,b,x,y;
int agree;

/* Initial configurations*/

no = 0;
for( a=0.0; a<= 1.0; a+=S) {
  for( b=0.0; b<=1.0; b+=S) {
    folge2x[0][no]=a;
         folge2y[0][no]=b;
    bpts1x[no] = a;
    bpts1y[no] = b;
    no++;
  }
}
folgen1[0] = 0; folgen2[0] = no; bnpts1 = no; bnpts2 = 0;
agree = 0;
srand48(seed);

/*Forward Simulation*/

while (agree == 0) {
```

```
printf("agree = %d \n",agree);
k = 1;
for (i=0; i < T; i++) {

/* Update second component */

n1 = Poisson(beta);
if( n1 > maxnumber ) {
printf("array overfull\n");
exit(1);
 }

 folgen2[k] = 0;
 if( n1 > 0 ) for(m=0; m<n1; m++) {
    x = drand48();
    y = drand48();
    j=0;
    while(
      (Dist^2(x,y,folge1x[k-1][j],folge1y[k-1][j])>S*S)
       && (j < folgen1[k-1] )) j++;
    if( j == folgen1[k-1] ) {
       folge2x[k][folgen2[k]] = x;
       folge2y[k][folgen2[k]] = y;
      folgen2[k]++;
    }
 }

/* Update first component */

   n1 = Poisson(beta);
 if( n1 > maxnumber ) {
printf("array overfull\n");
exit(1);
 }
 folgen1[k]    = 0;
 folgenr[k-1]  = 0;
 if( n1 > 0 )  for(m=0; m<n1; m++) {
    x = drand48();
```

```
    y = drand48();
    j=0;
    while(
      (Dist^2(x,y,folge2x[k][j],folge2y[k][j])>S*S)
       && (j < folgen2[k] )) j++;
    if( j < folgen2[k] ) {
      folgerx[k-1][folgenr[k-1]] = x;
      folgery[k-1][folgenr[k-1]] = y;
      folgenr[k-1]++;
    }
          if( j == folgen2[k] ) {
      folge1x[k][folgen1[k]] = x;
      folge1y[k][folgen1[k]] = y;
      folgen1[k]++;
      }
}
k++;
} /* end of for loop*/

/* Backward Simulation */

for (i=(T-1); i>=0; i--)
        {
Backward(bpts1x, bpts1y, bnpts1,
         bpts2x, bpts2y, bnpts2,
         folge1x[i], folge1y[i], folgen1[i],
            folge2x[i], folge2y[i], folgen2[i],
         folgerx[i], folgery[i], folgenr[i],
         S);

        }

if (bnpts1 == 0) agree = 1;

return(folge1x[T], folge1y[T], folge2x[T], folge2y[T])

}
```

References

[1] E. Abbé. Über Blutkörper-Zählung. *Jena Zeit. Med. Naturw.*, 13 (Neue Serie 6):98–105, 1879.

[2] D. Aldous. A random walk construction of uniform spanning trees and uniform labelled trees. *SIAM Journal on Discrete Mathematics*, 3:450–465, 1990.

[3] D. Aldous and P. Diaconis. Strong uniform times and finite random walks. *Advances in Applied Mathematics*, 8:493 – 501, 1987.

[4] A.J. Baddeley, M. Kerscher, K. Schladitz, and B.T. Scott. Estimating the J-function without egde correction. Technical Report 1997/25, Department of Mathematics, University of Western Australia, 1997.

[5] A.J. Baddeley and M.N.M. van Lieshout. Area-interaction processes. *Ann. Inst. Statist. Math.*, 47:601–619, 1995.

[6] G. Barnard. Contribution to the discussion of Bartlett. *Journal of the Royal Statistical Society*, B 25:294, 1963.

[7] O.E. Barndorff-Nielsen and D.R. Cox. *Asymptotic Techniques for the use in statistics.* Chapman & Hall, London, 1989.

[8] R.A. Becker, J.M. Chambers, and A.R. Wilks. *The New S Language: A programming environment for Data Analysis and Graphics.* Computer Science Series. Wadsworth & Brooks, Pacific Grove, 1988.

[9] T. Bedford and J. van den Berg. A remark on Van Lieshout and Baddeley's j-function for point processes. *Advances in Applied Probability, Stochastic Geometry and Applications*, 29:19–25, 1997.

[10] J. Besag and P.J. Diggle. Simple Monte Carlo tests for spatial pattern. *Applied Statistics*, 26:327–333, 1977.

[11] A.A. Borovkov and S.G. Foss. Stochastically recursive sequences and their generalizations. *Siberian Advances in Mathematics*, 2:16–81, 1992.

[12] A.A. Borovkov and S.G. Foss. Two ergodicity criteria for stochastically recursive sequences. *Acta Applic Math*, 34:125–134, 1994.

[13] A. Broder. Generating random spanning trees. In *30th Annual Symposium on Foundations of Computer Science*, pages 442–447, 1989.

[14] S.P. Brooks. Markov chain Monte Carlo method and its application. *The Statistician*, 47:69 – 100, 1998.

[15] S.P. Brooks and G.O. Roberts. Assessing convergence of Markov Chain Monte Carlo algorithms. *Statistics and Computing*, 1999. To appear, preprint is available from *WWW* URL: `http://www.stats.bris.ac.uk:81/MCMC`.

[16] K. Burdzy and W.S. Kendall. Efficient Markovian couplings: examples and counterexamples. Research report 331, Department of Statistics, University of Warwick, 1998. Submitted for publication.

[17] Y. Cai and W.S. Kendall. Perfect Simulation for correlated Poisson variables conditioned to be positive. Technical report, Department of Statistics, University of Warwick, 1998. In preparation.

[18] D.S. Carter and B.M. Prenter. Exponential Spaces and Counting Processes. *Zeitschrift für Wahrscheinlichkeitstheorie und verwandte Gebiete*, 21:1–19, 1972.

[19] J.T. Chayes, L. Chayes, and R. Kotecký. The analysis of the Widom-Rowlinson model by stochastic geometry models. *Communications in Mathematical Physics*, 172:551–569, 1995.

[20] J.N. Corcoran and R.L. Tweedie. Perfect Sampling for Harris recurrent Markov chains. Technical report, University of Colorado, 1998.

[21] S.P. Cowie. The cumulative frequency nearest-neighbour method for the identification of spatial patterns. Technical Report A10, Department of Geography, University of Bristol, 1967.

[22] M.K. Cowles and B.P. Carlin. Markov chain convergence diagnostics: a comparative review. *Journal of the American Statistical Association*, 91:883–904, 1996.

[23] D.R. Cox. Some statistical methods connected with series of events (with discussion). *Journal of the Royal Statistical Society*, B 17:129–164, 1955.

[24] D.R. Cox and V. Isham. *Point Processes.* Chapman & Hall, London, 1980.

[25] N.A.C. Cressie. *Statistics for spatial data.* John Wiley & Sons, New York, revised edition, 1993.

[26] M. Creutz. Confinement and critical dimensionality of space-time. *Physical Review Letters*, 43:553 – 556, 1979.

[27] D.J. Daley and D. Vere-Jones. *An introduction to the theory of point processes.* Springer Verlag, New York, 1988.

[28] A.P. Dawid. Conditional Independence in statistical theory (with discussion). *Journal of the Royal Statistical Society*, B 41:1–31, 1979.

[29] A.P. Dawid. Conditional Independence for statistical operations. *Annals of Statistics*, 8:598–617, 1980.

[30] Luc Devroye. *Non-uniform random variate generation.* Springer Verlag, New York, 1986.

[31] P. Diaconis and J.A. Fill. Strong stationary times via a new form of duality. *Annals of Probability*, 18(4):1483 – 1522, 1990.

[32] P. Diaconis and D. Freedman. Iterated Random Functions. Technical Report 511, Department of Statistics, University of Carlifornia at Berkely, 1998.

[33] P.J. Diggle. On parameter estimation and goodness-of-fit testing for spatial point patterns. *Biometrics*, 35:87–101, 1979.

[34] P.J. Diggle. Statistical methods for spatial point patterns in ecology. In *Spatial and Temporal Analysis in Ecology*, pages 95–150. International Co-operative Publishing House, Burtonsville, 1980.

[35] P.J. Diggle. *Statistical analysis of spatial point patterns.* Academic Press, London, 1983.

[36] W. Doeblin. Exposé de la theorie de chaînes simples constantes de markov à un nombre fini d'états. *Rev. Math. Union Interbalkanique*, 2:77–105, 1938.

[37] J.A. Fill. An interruptible algorithm for exact sampling via Markov Chains. *Annals of Applied Probability*, 8:131–162, 1998.

[38] J.A. Fill. The Move-To-Front rule: A case study for two perfect sampling algorithms. *Probability in the Engineering and Informational Sciences*, 12:283–302, 1998.

[39] J.A. Fill and M. Machida. Stochastic Monotonicity and Realizable Monotonicity. *WWW* URL: `http://www.mts.jhu.edu/~fill/`, 1998.

[40] C.M. Fortuin and P.W. Kasteleyn. On the random cluster model, I. Introduction and relation to other models. *Physica*, 57:536–564, 1972.

[41] S.G. Foss and R.L. Tweedie. Perfect simulation and backward coupling. *Stochastic models*, 14:187–203, 1998.

[42] D. Gamerman. *Markov Chain Monte Carlo*. Chapman & Hall, London, 1998.

[43] A.E. Gelfand and A.F.M. Smith. Sampling based approaches to calculating marginal densities. *Journal of the American Statistical Association*, 85:398 – 409, 1990.

[44] A. Gelman and D. Rubin. Inference from iterative simulation using multiple sequences. *Statist. Sci*, 7:457–472, 1992.

[45] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images. *IEEE Transaction of Pattern Analysis and Machine Intelligence*, 6:721–741, 1984.

[46] C. Geyer and J. Møller. Simulation procedures and likelihood inference for spatial birth and death processes. *Scandinavian Journal of Statistics*, 21:359–373, 1994.

[47] C.J. Geyer. Practical Markov chain Monte Carlo. *Statist. Sci.*, 7:473–511, 1992.

[48] W.R. Gilks, S. Richardson, and D.J. Spiegelhalter, editors. *Markov chain Monte Carlo in Practice.* Chapman & Hall, London, 1996.

[49] R.J. Glauber. Time-dependent statistics of the Ising model. *J. Math. Phys.*, 4:294–307, 1963.

[50] P.J. Green. Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika*, 82:711–732, 1995.

[51] G.R. Grimmett. *Percolation.* Springer Verlag, New York, 1989.

[52] O. Häggström and K. Nelander. Exact sampling from anti-monotone systems. Technical report, Chalmers University, Göteborg, 1998. To appear in *Statistica Neerlandica.*

[53] O. Häggström and K. Nelander. On exact simulation of Markov random fields using coupling from the past. Technical report, Chalmers University, Göteborg, 1998. To appear in *Scandinavian Journal of Statistics.*

[54] O. Häggström, M.N.M van Lieshout, and J. Møller. Characterisation results and Markov chain Monte Carlo algorithms including exact simulation for some spatial point processes. Technical Report R-96-2040, Aalborg Mathematics Department, 1996. To appear in *Bernoulli.*

[55] P. Hall. *Introduction to the theory of coverage processes.* John Wiley & Sons, New York, 1988.

[56] J.M. Hammersley, J.W.E. Lewis, and J.S. Rowlinson. Relationships between the multinomial and Poisson models of stochastic processes, and between the canonical and grand canonical ensembles in statistical mechanics, with illustrations and Monte Carlo methods for the penetrable spherres model of liquid-vapour equilibrium. *Sankhya: The Indian Journal of Statistics*, A: 37:457–491, 1975.

[57] W.K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57:97 – 109, 1970.

[58] D.M. Higdon. Auxiliary variable methods for Markov Chain Monte Carlo. Preprint, Institute of Statistics and Decision Sciences, Duke University, 1997.

[59] E. Ising. Beitrag zur Theorie des Ferromagnetismus. *Zeitschrift für Physik*, 31:253–258, 1925.

[60] S.D. Jacka and G.O. Roberts. On strong form of weak convergence. *Stochastic Process. Appl.*, 67:41–53, 1997.

[61] D. Jeulin. Morphological modeling of images by sequential random functions. *Signal processing*, 16:403–431, 1989.

[62] D. Jeulin. Random models for morphological analysis of powders. *Journal of Microscopy*, 172:13 – 21, 1993.

[63] D. Jeulin. Dead Leaves Model: From space tessellation to random functions. In D. Jeulin, editor, *Advances in Theory and Applications of Random Sets*, pages 137–156. World Scientific Publishing Company, Singapore, 1997.

[64] D. Jeulin, I. Terol Villabos, and A. Dubus. Morphological analysis of UO_2 Powder using a Dead Leaves Model. *Microsc. Microanal. Microstruct.*, 6:371 – 384, 1995.

[65] V.E. Johnson. Studying convergence of Markov chain Monte Carlo algorithms using coupled sample paths. *Journal of the American Ststistical Association*, 91:154–166, 1996.

[66] O. Kallenberg. An informal guide to the theory of conditioning in point processes. *Int. Statist. Rev.*, 52:151–164, 1984.

[67] T. Kamae, U. Krengel, and G.L. O'Brien. Stochastic inequalities on partially ordered spaces. *The Annals of Probability*, 5:899–912, 1977.

[68] D. Kandel, Y. Matias, R. Unger, and P. Winkler. Shuffling biological sequences. *Discrete Applied Mathematics*, 71:171–185, 1996.

[69] A.F. Karr. *Point Processes and their statistical inference.* Probability: Pure and applied. Marcel Deller Inc., New York, 1986.

[70] F.P. Kelly and B.D. Ripley. A note on Strauss's model for clustering. *Biometrika*, 63:357–360, 1976.

[71] W.S. Kendall. On some weighted Boolean Models. In D. Jeulin, editor, *Advances in Theory and Applications of Random Sets*, pages 105–120. World Scientific Publishing Company, Singapore, 1997.

[72] W.S. Kendall. Perfect simulation for spatial point processes. Research report 308, Department of Statistics, University of Warwick, 1997. To appear in Proc. ISI 51^{st} session, Istanbul (August 1997).

[73] W.S. Kendall. Perfect simulation for the area-interaction point process. In L. Accardi and C.C. Heyde, editors, *Probability Towards 2000*, pages 218–234, New York, 1998. Springer.

[74] W.S. Kendall and J. Møller. Perfect simulation of point processes using Metropolis-Hastings algorithms. In preparation, 1997.

[75] W.S. Kendall and E. Thönnes. Perfect simulation in stochastic geometry. Research report 323, Department of Statistics, University of Warwick, 1997. To appear in the *Journal of Pattern Recognition: Special Issue on Random Sets.*

[76] R. Kindermann and J.L. Snell. *Markov random fields and their applications.* American Mathematical Society, Providence, Rhode Island, 1980.

[77] W. Klein. Potts-model formulation of Continuum Percolation. *Physical Review*, B 26:2677–2678, 1993.

[78] D.E. Knuth. *The art of computer programming, Volume 2: Seminumerical Algorithms.* Addison-Wesley, Reading, 1969.

[79] C. Lantuéjoul. Conditional Simulation Of Object-Based Models. In D. Jeulin, editor, *Advances in Theory and Applications of Random Sets*, pages 271–288. World Scientific Publishing Company, Singapore, 1997.

[80] D.H. Lehmer. Mathematical methods in large-scale computing units. In *Proceedings of the second Symposium on Large-Scale Digital Computing Machinery*, pages 142 – 145. Harvard University Press, Cambridge, 1951.

[81] W. Lenz. Beitrag zum Verständnis der magnetischen Erscheinungen in festen Körpern. *Zeitschrift für Physik*, 21:613–615, 1920.

[82] P.A.W. Lewis, A.S. Goodman, and J.M. Miller. A pseudo-random number generator for the system 360. *IBM Systems Journal*, 8:136–145, 1969.

[83] T.G. Lewis and W.H. Payne. Generalized feedback shift register pseudo-random number algorithm. *J. Assoc. Comput. Mach*, 20:456 – 468, 1973.

[84] T. Lindvall. *Lectures on the coupling method.* Wiley Series in Probability and Mathematical Statistics. John Wiley & Sons, New York, 1992.

[85] H.W. Lotwick and B.W. Silverman. Convergence of spatial birth-and-death processes. *Math. Proc. Camb. Phil. Soc.*, 90:155–165, 1981.

[86] B. Matérn. Spatial variation. *Meddelanden fran statens Skogsforskningsinstitut*, 49:1–141, 1960.

[87] G. Matheron. *Random Sets and Integral Geometry.* John Wiley & Sons, New York, 1975.

[88] N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller, and E. Teller. Equations of state calculations by fast computing machines. *Journal of Chemical Physics*, 21:1087 – 1092, 1953.

[89] N. Metropolis and S. Ulam. The Monte Carlo method. *Journal of the American Statistical Association*, 44:335 – 341, 1949.

[90] S.P. Meyn and R.L. Tweedie. *Markov chains and stochastic stability.* Springer Verlag, New York, 1993.

[91] S.P. Meyn and R.L. Tweedie. Stability of Markovian processes II: Continuous-time processes and sampled chains. *Advances in Applied Probability*, 25:487–517, 1993.

[92] A. Mira and L. Tierney. On the use of auxiliary variables in Markov chain Monte Carlo sampling. Preprint, School of Statistics, University of Minnesota, 1997.

[93] I. Molchanov. *Statistics of the Boolean Model for Practitioners and Mathematicians.* Wiley & Sons, Chichester, 1997.

[94] J. Møller. On the rate of convergence of spatial birth-and-death processes. *Ann. Inst. Statist. Math.*, 41(3):565–581, 1989.

[95] J. Møller. Markov chain Monte Carlo and spatial point processes. In W.S. Kendall O.E. Barndorff-Nielsen and M.N.M. van Lieshout, editors, *Stochastic geometry, computation and likelihood*, New York, 1998. Proceedings Séminaire Européen de Statistique, Chapman & Hall. To appear.

[96] J. Møller and K. Schladitz. Extensions of Fill's algorithm for perfect simulation. Technical report, Aalborg Mathematics Department, 1998. Submitted for publication.

[97] D.J. Murdoch and P.J. Green. Exact sampling from a continuous state space. Technical report, Department of Mathematics, University of Bristol, 1997. To appear in Scandinavian Journal of Statistics.

[98] D.J. Murdoch and P.J. Green. Exact sampling for bayesian inference: towards general purpose algorithms. Technical report, Department of Mathematics, University of Bristol, 1998. To appear in 6th Valencia International Meeting on Bayesian statistics.

[99] D.J. Murdoch and J.S. Rosenthal. Efficient use of exact samples. Technical report, University of Toronto, 1998. Preprint is available from *WWW* URL: `http://www.stats.bris.ac.uk:81/MCMC`.

[100] D.J. Murdoch and J.S. Rosenthal. An extension of Fill's exact sampling algorithm to non-monotone chains. Technical report, University of Toronto, 1998. Preprint is available from *WWW* URL: `http://www.stats.bris.ac.uk:81/MCMC`.

[101] R.N. Neal. Markov Chain Monte Carlo Methods based on 'slicing' the density function. Technical Report No. 9722, Department of Satistics, University of Toronto, 1997.

[102] J. Neyman and E.L. Scott. Statistical approach to problems of cosmology. *Journal of the Royal Statistical Society*, B 20:1–43, 1958.

[103] X.X. Nguyen and H. Zessin. Integral and differential characterisation of Gibbs processes. *Mathematische Nachrichten*, 88:105–115, 1979.

[104] H. Niederreiter. *Random Number Generation and Quasi-Monte Carlo Methods.* CBMS-NSF Regional Conference Series in Applied Mathematics, Society for Industrial and Applied Mathematics, Philadelphia, 1992.

[105] E. Nummelin. *General Irreducible Markov chains and Non-negative Operators.* Cambridge University Press, Cambridge, 1984.

[106] C. Palm. Intensitätsschwankungen im Fernsprechverkehr. *Ericson Technik*, 44:1–189, 1943.

[107] F. Papangelou. The conditional intensity of general point processes and an application to line processes. *Zeit. Wahrsch.*, 28:207–226, 1974.

[108] R.B. Potts. Some generalized Order-Disorder Transformations. *Proceedings of the Cambridge Philosophical Society*, 48:106–109, 1952.

[109] C.J. Preston. *Gibbs States on countable sets.* Cambridge University Press, London, 1974.

[110] C.J. Preston. Spatial birth-and-death processes. *Bulletin of the International Statistical Institute*, 46:371–391, 1975.

[111] J.G. Propp and D.B. Wilson. Exact sampling with coupled Markov chains and applications to statistical mechanics. *Random Structures and Algorithms*, 9:223–252, 1996.

[112] J.G. Propp and D.B. Wilson. Coupling from the past: a user's guide. In D. Aldous and J.G. Propp, editors, *Microsurveys in Discrete Probability*, volume 41 of DIMACS Series in Discrete Mathematics and Theoretical Computer Science, pages 181–192. American Mathematical Society, Rhode Island, 1998.

[113] J.G. Propp and D.B. Wilson. How to get a perfectly random sample from a generic Markov chain and generate a random spanning tree of a directed graph. *Journal of Algorithms*, 27:170–217, 1998.

[114] R.-D. Reiss. *A course on Point Processes.* Springer Verlag, Berlin, 1993.

[115] B.D. Ripley. Modelling spatial patterns (with discussion). *Royal Statistical Soceity*, B 39:172 – 212, 1977.

[116] B.D. Ripley. Simulating spatial patterns: dependent samples from a multivariate density, Algorithm AS 137. *Applied Statistics*, 28:109 – 112, 1979.

[117] B.D. Ripley. *Spatial Statistics*. John Wiley & Sons, New York, 1981.

[118] B.D. Ripley. *Stochastic Simulation*. John Wiley & Sons, New York, 1987.

[119] B.D. Ripley. *Statistical Inference for Spatial Processes*. Cambridge University Press, Cambridge, 1988.

[120] B.D. Ripley and F.P. Kelly. Markov point processes. *Journal of the London Mathematical Society*, 15:188–192, 1977.

[121] G.O. Roberts and N.G. Polson. On the geometric convergence of the Gibbs sampler. *Journal of the Royal Statistical Society*, B 56:377–384, 1994.

[122] G.O. Roberts and J.S. Rosenthal. Convergence of slice sampler Markov chains. Preprint available from *WWW* URL: `http://www.stats.bris.ac.uk:81/MCMC`, 1997.

[123] G.O. Roberts and A.F.M. Smith. Simple conditions for the convergence of the Gibbs Sampler and the Metropolis-Hastings algorithm. *Stochastic Processes and their application*, 49:207 – 216, 1994.

[124] J.S Rosenthal. Minorization conditions and convergence rates for Markov chain Monte Carlo. *Journal of the American Statistical Association*, pages 558–566, 1995.

[125] J.S. Rowlinson. Penetrable spheres model of liquid vapour equilibrium. *Advances in Chemical Physics*, 41:1–57, 1980.

[126] J.S. Rowlinson. Probability densities for some one-dimensional problems in statistical mechanics. In *Disorder in physical systems*, pages 261–276. Clarendon Press, Oxford, 1990.

[127] L. Saloff-Coste. Simple examples of the use of Nash inequalities for finite Markov chains. In O. Barndorff-Nielsen, W.S. Kendall, and M.N.M van Lieshout, editors, *Stochastic Geometry: likelihood and computation*. Chapman & Hall, London, 1997. To appear.

[128] I. Saxl and J. Rataj. Spherical contact and nearest-neighbour distances in boolean cluster fields. *Acta Stereologica*, 15:91–96, 1996.

[129] H. Seidel. Über die Probabilitäten solcher Ereignisse, welche nur selten vorkommen, obgleich sie unbeschränkt oft möglich sind. *Sitzungsber. Math. Phys. Cl. Akad. Wiss. München*, pages 44–50, 1876.

[130] J. Serra. *Image Analysis and Mathematical Morphology*. Academic Press, London, 1982.

[131] I. M. Slivnyak. Some properties of stationary flows of homogenous random events. *Teor. Veroyat. Primen.*, 7:347–352, 1962. Translation in *Theory Prob. Appl.*, 7, 336-341.

[132] F. Spitzer. Random fields and interacting particle systems. M.A.A. Summer Seminar notes, 1971.

[133] D. Stoyan, W.S. Kendall, and J. Mecke. *Stochastic Geometry and its applications*. Wiley Series in Probability and Statistics. John Wiley & Sons, Chichester, second edition, 1995.

[134] D. Stoyan and H. Stoyan. *Fractals, random shapes and point fields*. John Wiley & Sons, Chichester, 1994.

[135] V. Strassen. The existence of probability measures with given marginals. *Ann Math. Statist.*, 36:423–439, 1965.

[136] D.J. Strauss. A model for clustering. *Biometrika*, 63:467–475, 1975.

[137] M. Sweeny. Monte Carlo study of wheighted percolation clusters relevant to the Potts models. *Physical Review*, B 27:4445–4455, 1983.

[138] R.H. Swendsen and J.S. Wang. Non-universal critical dynamics in Monte Carlo simulation. *Physical Review Letters*, 58:86–88, 1987.

[139] R.C. Tausworthe. Random numbers generated by linear recurrence modulo two. *Mathematical Computing*, 19:201 – 209, 1965.

[140] E. Thönnes. Perfect simulation of some Point Processes for the Impatient User. Technical Report 317, Department of Statistics, University of Warwick, 1997. To appear in *Advances in Applied Probability, Stochastic Geometry and Statistical Applications.*

[141] E. Thönnes and M.N.M. van Lieshout. A comparative study on the power of van Lieshout and Baddeley's J-function. Technical Report 334, Department of Statistics, University of Warwick, 1998. submitted for publication.

[142] H. Thorisson. Coupling methods in Probability Theory. *Scandinavian Journal of Statistics*, 22:159–182, 1995.

[143] L. Tierney. Markov chains for exploring posterior distributions. *Annals of Statistics*, 22:1701 – 1762, 1994.

[144] L. Tierney. Introduction to general state-space Markov chain theory. In D.J. Spiegelhalter W.R. Gilks, S. Richardson, editor, *Markov chain Monte Carlo in Practice*, pages 59 – 74. Chapman & Hall, London, 1996.

[145] M.N.M van Lieshout and A.J. Baddeley. A nonparametric measure of spatial interaction in point patterns. *Statistica Neerlandica*, 3:344–361, 1996.

[146] M.N.M van Lieshout and A.J. Baddeley. Indices of dependence between types in multivariate point patterns. Technical Report UWA 97/17, University of Western Australia, 1997. To appear in *Scandinavian Journal of Statistics.*

[147] B. Widom and J.S. Rowlinson. A new model for the study of liquid-vapor phase transitions. *Journal of Chemical Physics*, 52:1670–1684, 1970.

[148] D. Wilson. Annotated Bibliography of perfectly random sampling with markov chains. In D. Aldous and J. Propp, editors, *Microsurveys in Discrete Probability*, volume 41 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, Rhode Island, 1998. American Mathematical Society. Updated versions on `http://dimacs.rutgers.edu/~dbwilson/exact.html`.

[149] G. Winkler. *Image Analysis, Random Fields and Dynamic Monte Carlo Methods.* Springer, Berlin, 1995.

[150] F. Y. Wu. The Potts model. *Reviews of Modern Physics*, 54:235–268, 1982.

4